Available online at www.sciencedirect.com

SCIENCE DIRECT®

Artificial Intelligence 170 (2006) 232–297

Artificial Intelligence

www.elsevier.com/locate/artint

Flexible diagnosis of discrete-event systems by similarity-based reasoning techniques

Gianfranco Lamperti *, Marina Zanella

Dipartimento di Elettronica per l'Automazione, Università di Brescia, via Branze 38, 25123 Brescia, Italy

Received 23 December 2003; received in revised form 3 August 2005; accepted 29 August 2005

Available online 29 September 2005

Abstract

Diagnosis of discrete-event systems (DESs) may be improved by knowledge-compilation techniques, where a large amount of model-based reasoning is anticipated off-line, by simulating the behavior of the system and generating suitable data structures (compiled knowledge) embedding diagnostic information. This knowledge is exploited on-line, based on the observation of the system behavior, so as to generate the set of candidate diagnoses (problem solution). This paper makes a step forward: the solution of a diagnostic problem is supported by the solution of another problem, provided the two problems are somewhat similar. Reuse of model-based reasoning is thus achieved by exploiting the diagnostic knowledge yielded for solving previous problems. The technique still works when the available knowledge does not fit the extent of the system, but only a partition of it, that is, when solutions are available for subsystems only. In this case, the fragmented knowledge is exploited in a modular way, where redundant computation is avoided. Similarity-based diagnosis is meant for large-scale DESs, where the degree of similarity among subsystems is high and stringent time constraints on the diagnosis response is a first-class requirement.

Keywords: Diagnosis; Model-based reasoning; Similarity-based reasoning; Discrete-event systems; Communicating automata; Knowledge compilation; Subsumption; Reusability; Uncertainty

* Corresponding author.
E-mail addresses: lamperti@ing.unibs.it (G. Lamperti), zanella@ing.unibs.it (M. Zanella).

0004-3702/$ – see front matter
doi:10.1016/j.artint.2005.08.002

1. Introduction

Discrete-event systems (DESs) [1] are dynamic systems with discrete inputs and outputs, whose favorite behavioral models are finite automata. Since, at some level of abstraction, most real-world systems can be viewed as DESs and reasoning about discrete models is easier than about continuous ones, from the middle '90s the task of diagnosis of DESs has been receiving an increasing interest from both the Artificial Intelligence [2–6] and the Automatic Control communities [7,8].

Diagnosing a system means computing its *candidate diagnoses*, each of which is a set of faults that explains the observation collected during the system operation. In the general case, the specific faults of a DES cannot be inferred without finding out what has happened to the system [9]. This way, the system evolutions complying with the observation, be they called histories [10], situation histories or narratives [11], paths [12], or trajectories [13], become a product of the diagnostic reasoning.

Determining the system evolutions is computationally expensive (see [14] about the difficulties of the diagnoser approach [15,16], or the worst case computational complexity analysis in [10], or the discussion in [17]). This is why most approaches exploit a trade-off between *off-line* and *on-line* computation: some kind of knowledge, implicit in the models of the structure and behavior of the system, is compiled off-line in order to speed up on-line processing.

This paper applies *knowledge compilation* and *similarity-based reasoning* to the active system approach [10,18], which deals with diagnosis of a class of DESs, called *active systems* [19]. According to other approaches in the literature, compiled knowledge is produced once and for all before any diagnostic problem is considered, then such a knowledge is exploited several times on-line, and it never changes. This paper suggests how to increase the (possibly null) compiled knowledge generated beforehand by progressively adding to it the (intermediate and/or final) graph-based data produced on-line when solving diagnostic problems. The proposed shift of perspective charges the diagnostic process with further responsibilities: exploiting available knowledge (if any) and generating new knowledge.

Flexibility groups a number of requirements for the diagnostic method, including *reusability* and *modularity*. Reusability applies both to component models and to any piece of knowledge produced either off-line or on-line. Modularity is the ability to recursively decompose a diagnostic problem into subproblems, to obtain a hierarchy where independent problems can be solved in parallel. Modularity characterizes also the previous active system approach [10,18,20]: however, the challenge faced in this paper is to get modularity cooperate with reusability by supporting knowledge compilation and be supported by it. This means that, on the one hand, a problem is not further decomposed if there already exists a piece of knowledge that can be exploited for its solution and, on the other, the solution process of each problem brings to the generation of persistent knowledge to be exploited in the future.

Intuitively, the proposed technique achieves larger computational savings when dealing with diagnostic problems characterized by 'regular' features. The first feature, the system to be diagnosed, is regular if it contains several instances of isomorphic subsystems, each of which is regular itself. The second feature is observability, which is regular if

what is observable in a subsystem can be obtained by restricting (and possibly renaming) what is observable in an isomorphic one. Another feature is abnormality, that is the faults we are interested in: this is regular if what is reckoned as faulty in a subsystem can be obtained by restricting (and possibly renaming) what is considered faulty in an isomorphic one. The last problem feature, the observation, is regular if the observation of a subsystem can be obtained by restricting (and possibly renaming) that of an isomorphic one.

The paper is organized as follows. Section 2 introduces a sample application domain as a concrete reference for the given examples. Section 3 lists the requirements that inspired the diagnostic technique. Section 4 formalizes the primitives for system modeling. Section 5 introduces the notion of a behavior space, wherein system evolutions are confined. Section 6 gives a formal definition of diagnostic problem and relevant solution. Section 7 sets the conditions under which similarity-based reasoning is applicable. Section 8 outlines a technique for solving diagnostic problems when no similarity-based reasoning is applicable. Section 9 focuses on off-line knowledge compilation, a preprocessing technique for generating diagnostic knowledge to be exploited in problem solving. Section 10 formalizes the notion of diagnosis by similarity-based reasoning. Section 11 extends the similarity-based diagnostic technique to problems supported by fragmented knowledge. Section 12 discusses the proposed method and compares it with other approaches in the literature. Conclusions are drawn in Section 13. Appendix A provides the complexity analyses of some algorithms belonging to the method and the proof sketches of the propositions stated in Sections 7–11.

2. Device network

We consider a sample application domain involving networks of electrical devices. Each device is protected by two breakers that are commanded by a protection. The protection is designed to detect dangerous conditions. Typically, if a short circuit affects the device, the protection is expected to trip the two breakers to open. In a simplified view, the network is represented by a series of devices, each one associated with a protection, as displayed in Fig. 1, where devices $D_1 \ldots D_4$ are protected by protections $p_1 \ldots p_4$. For instance, p_2 controls D_2 by operating breakers b_{21} and b_{22}. In normal (correct) behavior, both breakers are expected to open when tripped by the protection. However, the protection system may exhibit an abnormal (faulty) behavior, for example, one breaker or both may not open when required. In such a case, each faulty breaker informs the protection about its own misbehavior. Then, the protection sends a request of recovery actions to the neighboring protections, which will operate their own breakers appropriately. For example, if p_2 operates b_{21} and b_{22} and the latter is faulty, then p_2 will send a signal to p_3, which is supposed

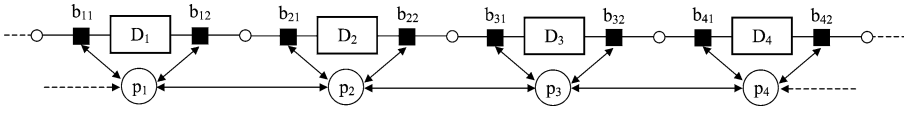

Fig. 1. Device network.

to command b_{32}. A recovery action may be faulty on its turn. The protection system is designed to propagate the recovery request until the tripped breaker opens correctly. When the protection system is reacting, a subset of the occurring events are visible to the operator in a control room who is in charge of monitoring the behavior of the network and, possibly, to issue explicit commands so as to minimize the extent of the isolated subnetwork. The localization of the short circuit and the identification of the faulty breakers may be impractical in real contexts, especially when the extent of the isolation spans several devices and the operator is required to take recovery actions within stringent time constraints. On the one hand, there is the problem of observability: the observable events generated during the reaction of the protection system are generally incomplete and uncertain in nature. On the other, whatever the 'quality' of the observation, it is impractical for the operator to reason on the observations so as to make consistent hypotheses on the behavior of the system and, eventually, to establish the shorted device and the faulty breakers.

3. Flexibility requirements

The diagnostic techniques proposed in this paper are inspired by six general *flexibility requirements*. Flexibility concerns all the three elements of the diagnostic task, namely, the system modeling (Requirement 1), the formulation of the diagnostic problem (Requirement 2), and the solution of the problem (Requirements 3–6).

Requirement 1 (Separation of concerns). *Both observability and abnormality properties of a system shall not be statically bound to the system model*: *they shall be dynamically bound to the actual diagnostic problem.*

The complete modeling of a DES encompasses both normal and abnormal behavior. Typically, the model of each component of the system is defined by an automaton whose state transitions can be either normal or abnormal (faulty). Besides, each transition may be either observable or unobservable (silent). It is commonplace embedding observability specifications within the model of the component. By contrast, owing to both practical and formal reasons, the flexible approach defers such a binding to problem-formulation time. Likewise, what is to be considered abnormal is usually part of the model description. This static binding exhibits however two shortcomings. On the one hand, what is to be considered a faulty behavior might be a matter of opinion. On the other, even when the separation between normal and abnormal behavior is statically clear, there is the additional problem of the granularity of the diagnostic information. A breaker, for example, may be faulty in different ways, including *stuck-to-close* and *unopen*, denoting permanent and transient faults, respectively. However, such a fine-grained diagnostic information might be inappropriate or useless to the operator, who might be at most interested in whether the breaker misbehaved or not during the reaction. In a coarser-grained definition, the diagnosis might embody the set of faulty components rather than the occurrences of the specific faulty events.

Requirement 2 (Uncertainty). *The diagnostic problem shall be uncertain in nature.*

A major element of the diagnostic problem is the system observation. Ideally, the observation relevant to a system reaction is a sequence of observable events, in accordance with the observability property specified in the diagnostic problem. Such a sequence is the trace of the observable transitions that are part of the system reaction. However, as outlined in [20], the linearity of the observation turns out to be an over-assumption in real application domains, where the system is large and distributed, and the communication channels are multiple and subjected to noise. Therefore, a system observation is no longer a list of observable events but rather a DAG, where nodes are uncertain observable events, while edges denote the partial ordering among the different pieces of the observation.[1]

Requirement 3 (Preprocessing). *The compositional model of the system shall be compiled off-line to generate diagnostic-oriented knowledge aimed at speeding up on-line diagnosis.*

The compositional approach to system modeling requires the specification of the topology of the system in terms of components and links among them, along with the model of each component (a communicating automaton), and the model of each link (typically, a queue). Solving a diagnostic problem amounts to reasoning on the system observation and the compositional model of the system, so as to reconstruct the system evolution and find out possible misbehaviors. Thus, without any off-line preprocessing, the diagnostic task requires the reconstruction of the system evolution and the subsequent distillation of the candidate diagnoses. Such tasks can be dramatically time-consuming in real applications. A considerable alleviation of the on-line diagnostic task may be pursued by performing as much as possible off-line preprocessing that is exploitable by on-line reconstruction/distillation.

Requirement 4 (Pattern-matching). *Once model compilation has generated knowledge off-line, on-line diagnosis shall be confined to a pattern-matching activity.*

When a diagnostic problem is formulated on-line, there is no need for model-based reasoning, as the constraints enforced by the model are implicitly incorporated in the compiled knowledge. Essentially, the additional constraints imposed on-line by the diagnostic problem are confined to the system observation. Since no information other than observable events is relevant for pattern matching, the compiled knowledge can be manipulated so as to represent the regular language of the sentences of the subsystem, each sentence being a sequence of observable events.

Requirement 5 (Modularity). *The compiled diagnostic knowledge shall be exploitable even when the extent of the system exceeds the domain of the knowledge.*

Preprocessing is devoted to the generation of diagnostic knowledge for subparts of a system, as the preprocessing of the whole system is assumed to be impractical in large-scale applications. The extent of a given diagnostic problem is generally a subset of the

[1] Requirement 2 refers to a specific meaning of uncertainty, which is focused on the notion of a temporal observation introduced in [20], rather than on system modeling based on probability.

whole system, which possibly encompasses several subparts for which diagnostic knowledge is available. In this case, on-line diagnosis via pattern-matching cannot be pursued. However, we require that the solution of the diagnostic problem be supported by the modular exploitation of the available partial knowledge.

Requirement 6 (Reusability). *The knowledge generated on-line for solving a diagnostic problem shall be possibly reused when solving a different diagnostic problem.*

The solution of a diagnostic problem $\wp$ is bound to generate additional knowledge specific for $\wp$. Generally speaking, such knowledge is useless for the solution of a different diagnostic problem $\wp'$. However, chances are, $\wp'$ is somewhat similar to $\wp$. In this case, the knowledge generated for $\wp$ might be suitable for solving $\wp'$ as well. This way, two sorts of compiled knowledge are expected to support on-line diagnostic-problem solving:

(1) General-purpose diagnostic knowledge compiled off-line independently of any specific observation;
(2) Special-purpose diagnostic knowledge generated on-line for solving actual diagnostic problems, based on specific observations.

The reusability requirement refers to the latter. Knowledge reuse requires the diagnostic engine to store the appropriate information in a knowledge catalog along with the relevant diagnostic-problem specification.

4. System modeling

In this section, we define the formal primitives for modeling a system. This is not the unique way in which a DES can be modeled, as several different modeling frameworks are proposed in the literature. However, it is essential to make some basic choices in order for the diagnostic technique to be clearly specified. We adopt compositional modeling, where the behavior of the system is implicitly defined by its topology and the behaviors of its components.

A system is a network of *components* that are connected to one another through *links*. Each component is completely modeled by a *communicating automaton* that reacts to events either coming from the external world or from neighboring components through links. Formally, the component model is a 6-tuple $(\mathbf{S}, \mathbf{E}_{\text{in}}, \mathbf{I}, \mathbf{E}_{\text{out}}, \mathbf{O}, \mathbf{T})$, where $\mathbf{S}$ is the set of *states*, $\mathbf{E}_{\text{in}}$ the set of *input events*, $\mathbf{I}$ the set of *input terminals*, $\mathbf{E}_{\text{out}}$ the set of *output events*, $\mathbf{O}$ the set of *output terminals*, and $\mathbf{T}$ the nondeterministic *transition function* $\mathbf{T} : \mathbf{S} \times \mathbf{E}_{\text{in}} \times \mathbf{I} \times 2^{\mathbf{E}_{\text{out}} \times \mathbf{O}} \mapsto 2^{\mathbf{S}}$. A transition $T \in \mathbf{T}$, from state S to state S', that is triggered by an event e at input terminal I, and generates the events $e_1, \ldots, e_k$ at output terminals $O_1, \ldots, O_k$, respectively, is denoted by[2]

$$T = S \xrightarrow[(e_1, O_1), \ldots, (e_k, O_k)]{(e, I)} S'. \tag{1}$$

[2] Although the transition function is nondeterministic, in practice, a transition can be identified based on the association with one possible state S'.

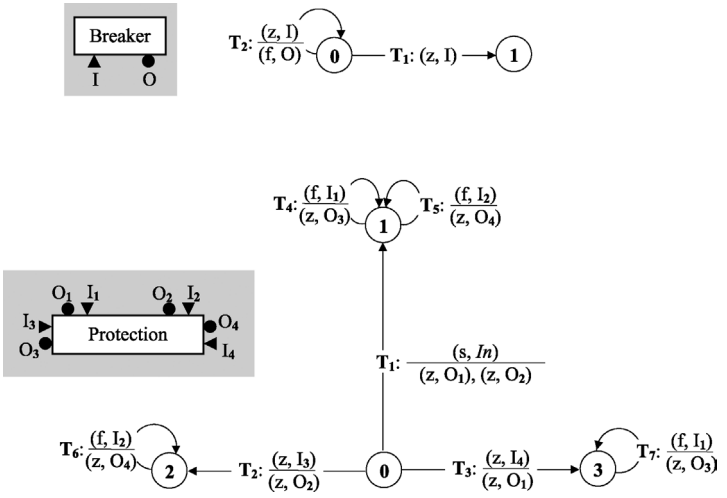

Fig. 2. Component models.

All components are implicitly equipped with an *In* terminal, the *standard input*, which is meant for events coming from the external world.

Links are the means to store the events exchanged between components. A *link model* is a triple (I, O, K), where I is the *input terminal*, O the *output terminal*, and K the *capacity*. The latter is the (finite) maximum number of storable events. A link is an instantiation of a link model. When the link is full (the number of stored events equals its capacity), the attempt to insert a new event into the link results in the loss of the event. Dually, no event can be consumed when the link is empty. Events are queued into the link and consumed based on a FIFO policy. The sequence of events stored in the link is a *configuration* of the link.

A *system model* is a pair $(\mathbf{C}^{\mathrm{m}}, \mathbf{L}^{\mathrm{m}})$, where $\mathbf{C}^{\mathrm{m}}$ is the multiset of component models and $\mathbf{L}^{\mathrm{m}}$ the multiset of link models. A *system* $\Psi = (\mathbf{C}, \mathbf{L})$ is an instantiation of a system model,[3] where $\mathbf{C}$ is the set of components instantiating $\mathbf{C}^{\mathrm{m}}$ and $\mathbf{L}$ the set of links instantiating $\mathbf{L}^{\mathrm{m}}$. The *dangling terminals* of Ψ is the set of terminals of components in $\mathbf{C}$ that are not connected with any link in $\mathbf{L}$. No assumption is made on events at dangling terminals. A *subsystem* of Ψ is a system $\Psi' = (\mathbf{C}', \mathbf{L}')$, where $\mathbf{C}' \subseteq \mathbf{C}$ and $\mathbf{L}' \subseteq \mathbf{L}$ is the subset of links in $\mathbf{L}$ that connect (in the same way as in Ψ) components in $\mathbf{C}'$.

Example 1. Displayed in Fig. 2 are the models *Breaker* (top) and *Protection* (bottom), relevant to the protection system outlined in Fig. 1. Each model is depicted by the set of terminals (left) and the communicating automaton (right). Consider the model of the

[3] Note how the real physical system is not considered in the formalization, as both the system model and the system are in fact different modeling abstractions, rather than physical entities. However, the distinction between the system model and the system is meant to define several different systems in terms of the same topological and behavioral pattern. This evokes the approach adopted in object-oriented languages, where the same class is instantiated by several objects.

breaker. Within the shaded box, the input terminal I is depicted as a triangle, while the output terminal O is represented as a bullet. The relevant automaton (on the right) incorporates two states, marked by 0 (closed) and 1 (open), respectively, and two transitions, T_1 and T_2, represented as arrows between states. When the breaker is closed (state 0), either transition T_1 or T_2 is nondeterministically triggered by event z on input terminal I. T_1 moves the breaker to state 1 (open) without generating any output event. T_2, instead, keeps the state of the breaker unchanged (closed), whilst generating event f at output terminal O. Intuitively, this is an abnormal transition,[4] as the breaker is supposed to open when triggered, which is not the case for T_2.

The model of the protection embodies four input terminals, $I_1 \dots I_4$, and four output terminals, $O_1 \dots O_4$. Terminals O_1 and I_1 are meant for connection with the breaker on the left of the device, while terminals O_2 and I_2 are for the communication with the breaker on the right. I_3 and O_3 allow the protection to exchange events with the neighboring protection on the left. The same applies for I_4 and O_4, which are a means to communicate with the adjacent protection on the right. The corresponding automaton involves four states, marked by $0 \dots 3$, and seven transitions, $T_1 \dots T_7$. State 0 stands for ordinary condition, when no short circuit has occurred.

The occurrence of a short circuit on the protected device is signaled by event s at the standard input *In*, which triggers transition T_1. Such a transition moves the protection to state 1 by generating event z at both output terminals O_1 and O_2, thus commanding the two breakers to open. In state 1, the protection may receive event f either at terminal I_1 or I_2, meaning that the relevant breaker failed to open. This triggers either transition T_4 or T_5, respectively, each of which generates event z at output terminals O_3 and O_4, respectively. This reaction complies with the recovery actions described in Section 2, where the intervention of a breaker in the adjacent protection is required.

When a protection receives a request of recovery from a neighboring protection, it performs a transition from state 0 to either 2 or 3, depending on whether the request comes from the left (T_2) or from the right (T_3), respectively. So, input event (z, I_3) causes T_2 to generate (z, O_2), that is, a command to the breaker on the right. In state 2, since even this breaker may in turn be faulty, the occurrence of event (f, I_2) triggers transition T_6 that, similarly to T_5, propagates the recovery request to the right-hand side protection. A symmetric behavior is defined in state 3.

Depicted in Fig. 3 is the topology of a system ξ, that integrates the three components protecting a generic device, namely protection p and breakers b_1 and b_2. The protection is connected with the breakers by means of links $L_1 \dots L_4$. We assume that all links share the same model, with capacity $K = 1$. Note the dangling terminals I_3, O_3, I_4, and O_4. This means that transitions of protection p may be triggered by events on I_3 and I_4 as well as by the short circuit external event. System ξ is the abstraction of a subpart of the protection system of a device network. Larger subparts of the network may be assembled by connecting instances of ξ by means of links among protections.

[4] According to Requirement 1, what is to be considered abnormal and the mode in which abnormal behavior is to be encoded in the diagnostic output is not embedded within the component model but, rather, in the actual diagnostic problem, as detailed in Section 6.

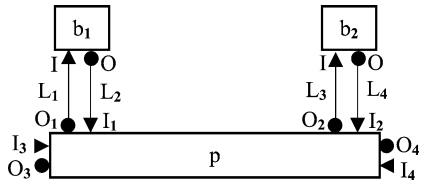

Fig. 3. System ξ.

System isomorphism

Flexible diagnosis is supported by the notion of isomorphism between systems. Two systems ψ and ψ' are *isomorphic*, denoted $\psi \doteq \psi'$, if they are instantiations of the same system model. Intuitively, the isomorphism between ψ and ψ' entails a matching between the relevant topologies, where each component C of ψ corresponds to one and only one component C' of ψ'. Within the isomorphism, the correspondence between C and C' is denoted by $C \between C'$. The same relationship holds for links, namely $L \between L'$, and for transitions, $T \between T'$, where T and T' are transitions of C and C', respectively. More precisely, $T \between T'$ is grounded on the relationship $C \between C'$. In fact, being corresponding components, C and C' are instances of the same component model, thereby sharing the same transition function $\mathbf{T}$. Thus, $T \between T'$ iff T and T' are the same transition in $\mathbf{T}$.

5. Behavior space

Once a system Ψ has been specified by compositional modeling, it may evolve only within a confined space. A *system state* is a pair $\sigma = (\mathbb{S}, \mathbb{L})$ where $\mathbb{S}$ is a record of the states of the components in Ψ, while $\mathbb{L}$ is a record of the configurations of the links in Ψ. Initially, a system Ψ is in a *quiescent* state Ψ_0, wherein all links are empty. Upon the arrival of an event from the external environment, Ψ becomes *reacting*, thereby making a series of system transitions, namely a *history* of Ψ. Due to asynchronism, each system transition is the transition of one component in Ψ. The whole set of possible evolutions of Ψ from the initial state Ψ_0 are specified by a *behavior space*. Formally, a behavior space is a finite automaton

$$Bhv(\Psi, \Psi_0) = (\Sigma, \mathbf{E}, \mathbf{T}, \Psi_0, \mathbf{Q}) \tag{2}$$

where Σ is the set of states of Ψ reachable from Ψ_0, $\mathbf{E}$ is the set of relevant component transitions, $\mathbf{T}$ is the transition function $\mathbf{T}: \Sigma \times \mathbf{E} \mapsto \Sigma$ such that $\sigma \xrightarrow{T} \sigma' \in \mathbf{T}$ iff σ' is the state of Ψ reached from σ through the component transition T, and $\mathbf{Q}$ is the set of quiescent states,

$$\mathbf{Q} = \big\{\sigma \mid \sigma \in \Sigma, \sigma = (\mathbb{S}, \mathbb{L}), \text{ where each link in } \mathbb{L} \text{ is empty}\big\}. \tag{3}$$

Since the behavior space is a finite automaton, the relevant *regular language* [21], $Lang(Bhv(\Psi, \Psi_0))$, is the (possibly unbounded) set of histories of $Bhv(\Psi, \Psi_0)$, each history being a phrase of such a language.

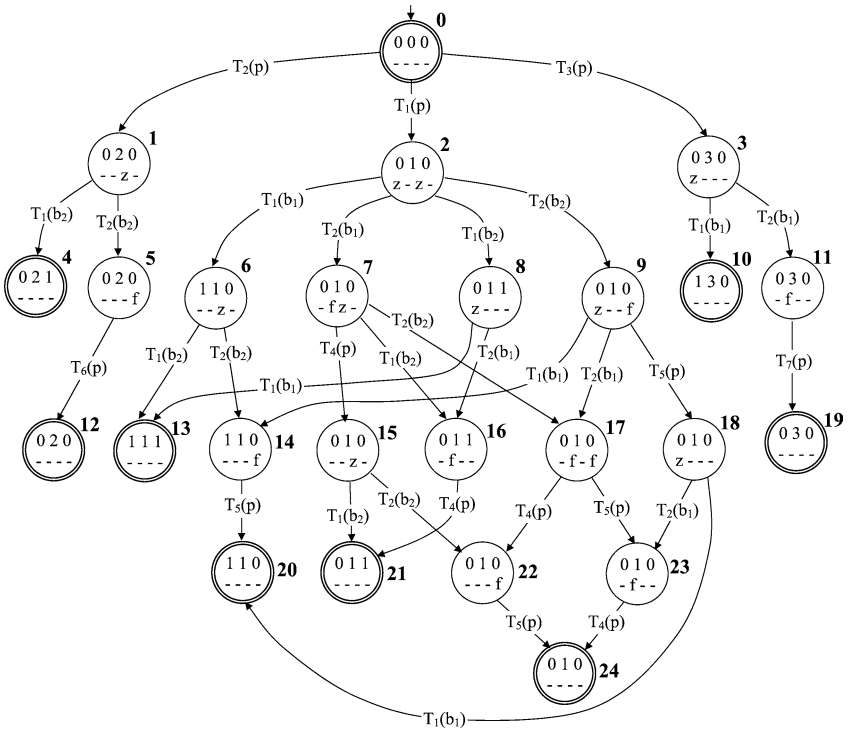

Fig. 4. Behavior space of ξ.

Example 2. Shown in Fig. 4 is the behavior space relevant to system ξ depicted in Fig. 3, where the initial (quiescent) state is $\xi_0 = (\mathbb{S}_0, \mathbb{L}_0)$, where $\mathbb{S}_0 = (0, 0, 0)$. Quiescent nodes are double circled. In each node, the record $\mathbb{S}$ of the component states for b_1, p, and b_2 is on the top, while the record $\mathbb{L}$ of queues of events within links $L_1 \ldots L_4$ is on the bottom. Since at most one event is stored in each link ($K = 1$), each configuration of the link can be expressed by either the label of the event or a dash, the latter denoting the empty link. Nodes are numbered. For instance, in node 7 both breakers are closed (state 0 of the breaker model), while the protection has commanded the breakers to open (state 1 of the protection model). Besides, links L_1 and L_4 are empty; instead, L_2 incorporates event f (meaning that b_1 has failed to open) while L_3 contains event z, meaning that b_2 has not yet reacted to the protection command. Each edge is marked by a transition followed by the name of the relevant component. Note how ξ becomes reacting upon the occurrence of an external event, either from the standard input (triggering transition $T_1(p)$) or from a dangling terminal (triggering either $T_2(p)$ or $T_3(p)$). Each (possibly empty) path from the initial state to a quiescent state is a history of ξ. A history $h(\xi)$ is identified by the sequence of labels (component transitions) marking the edges on such a path, as for instance, $h(\xi) = \langle T_1(p), T_2(b_1), T_1(b_2), T_4(p)\rangle$. The transitions in $h(\xi)$ leads us to the following scenario: (1) a short circuit occurs on the device protected by ξ, hence protection p commands both breakers b_1 and b_2 to open, (2) b_1 fails to open, (3) b_2 opens correctly, and (4) p

asks the neighboring protection on the left a recovery action. A peculiarity of the behavior space in Fig. 4 is acyclicity, so that the number of histories is finite. Generally speaking, however, a behavior space may include cycles, encompassing an unbounded number of histories. Another peculiarity is that each history includes only two quiescent states (initial and final). In the general case (typically, but not necessarily, when the graph is cyclic), histories may contain several quiescent states.

History restriction

Let h be a history of a system Ψ, and Ψ' a subsystem of Ψ. The *restriction* of h on Ψ', denoted $h_{\langle\Psi'\rangle}$, is a history for Ψ' obtained by removing from h the transitions that are not in Ψ', maintaining the relative order of the remaining transitions.

6. Diagnostic problem

The ultimate task of diagnosis is the solution of diagnostic problems. In our framework, a diagnostic problem is relevant to a system reaction. Solving a diagnostic problem amounts to finding out possible misbehaviors in the system reaction. A diagnostic problem is expressed by means of some sort of information about the system to be diagnosed and some clues on the system reaction. Formally, a diagnostic problem $\wp$ for a system Ψ is a 5-tuple

$$\wp(\Psi) = (\Psi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K}) \tag{4}$$

where:

(1) Ψ_0 is the *initial state* of Ψ, that is, the state of Ψ when the reaction started;
(2) $\mathcal{V}$ is the *viewer* (observer), with specific visibility properties;
(3) $\mathcal{O}$ is the *temporal observation* of the system generated during the reaction, as perceived by viewer $\mathcal{V}$;
(4) $\mathcal{R}$ is the *ruler*, which establishes what behavior is to be considered faulty and the granularity of the diagnosis;
(5) $\mathcal{K}$ is the *knowledge* of the system, including at least the compositional model of Ψ and, possibly, additional *compiled knowledge*.

Depending on knowledge $\mathcal{K}$, we distinguish two classes of diagnostic problems:

- *Model-based* diagnostic problems, when $\mathcal{K}$ is restricted to the compositional model of Ψ, that is, when no compiled knowledge is available;
- *Similarity-based* diagnostic problems, when $\mathcal{K}$ embodies compiled knowledge.

The solution of a similarity-based problem is virtually more efficient than that of a model-based problem, since part of the model-based reasoning necessary for solving the problem is somehow codified in $\mathcal{K}$.

6.1. Viewer

A viewer establishes at diagnostic-problem time what component transitions are somewhat visible, as well as the specific observable label for each of them. Formally, the viewer $\mathcal{V}$ of a diagnostic problem $\wp(\Psi)$ is a partial mapping between the set $\mathbf{T}$ of component transitions in Ψ and a set of labels $\boldsymbol{\Omega}$ (the observable events), $\mathcal{V}: \mathbf{T} \hookrightarrow \boldsymbol{\Omega}$. If $(T, \omega) \in \mathcal{V}$, then T is *visible*, otherwise T is *silent*.

This definition is general in nature: neither lexical constraints are imposed on the labels in $\boldsymbol{\Omega}$, nor special rules are requested for the mapping. In particular, the same label may be associated with one or more transitions. A label in $\boldsymbol{\Omega}$ that is associated by $\mathcal{V}$ with several transitions in $\mathbf{T}$ is *ambiguous*, and, likewise, $\mathcal{V}$ is ambiguous. The ambiguity of a viewer is reflected on the diagnosability degree of the system, as the ambiguous label prevents the binding with a specific component transition, thereby weakening the problem constraints. When $\mathcal{V} = \emptyset$, $\mathcal{V}$ is *blind*. As such, a blind viewer is incapable of observing any component transition at all. When the viewer is blind, we have a *blind diagnostic problem*.

Example 3. Considering the system ξ depicted in Fig. 3, along with the relevant component models outlined in Fig. 2, a possible (non-ambiguous) viewer $\mathcal{V}$ for a diagnostic problem $\wp(\xi)$ is defined for $\boldsymbol{\Omega} = \{o_1, o_2, l, r\}$ as follows:

$$\mathcal{V} = \big\{\big(T_1(b_1), o_1\big), \big(T_1(b_2), o_2\big), \big(T_2(p), l\big), \big(T_3(p), r\big)\big\}.$$

An ambiguous viewer $\mathcal{V}'$ for $\wp(\xi)$ might involve the ambiguous label o for both breakers, namely $\mathcal{V}' = \{(T_1(b_1), o), (T_1(b_2), o), (T_2(p), l), (T_3(p), r)\}$.

Viewer restriction

Let $\mathcal{V}$ be a viewer for a system Ψ, and Ψ' a subsystem of Ψ. The *restriction* of $\mathcal{V}$ on Ψ', denoted $\mathcal{V}_{\langle\Psi'\rangle}$, is a viewer for Ψ' defined as follows:

$$\mathcal{V}_{\langle\Psi'\rangle} = \big\{(T, \ell) \mid (T, \ell) \in \mathcal{V}, T \text{ is relevant to a component in } \Psi'\big\}. \tag{5}$$

Example 4. With reference to viewer $\mathcal{V}$ for system ξ defined in Example 3, consider the subsystem ξ' of ξ, which is composed of components b_1 and p only. The restriction of $\mathcal{V}$ on ξ' will be $\mathcal{V}_{\langle\xi'\rangle} = \{(T_1(b_1), o_1), (T_2(p), l), (T_3(p), r)\}$.

6.2. Temporal observation

A temporal observation $\mathcal{O}$ of a system Ψ is the set of observable events relevant to a reaction of Ψ, as perceived by a viewer, typically under uncertainty conditions, as expected by Requirement 2 and detailed in [20]. Let Λ be a finite domain of labels, including the *null* label ε. A temporal observation is a (not necessarily connected) DAG $(\mathbf{N}, \mathbf{E})$, where $\mathbf{N}$ is the set of nodes, each $N \in \mathbf{N}$ being marked with a non-empty subset of Λ, and $\mathbf{E}: \mathbf{N} \mapsto 2^{\mathbf{N}}$ is the set of edges. The '$\prec$' *temporal precedence* relationship among nodes of the graph is defined as follows:

(1) If $N \mapsto N' \in \mathbf{E}$ then $N \prec N'$;

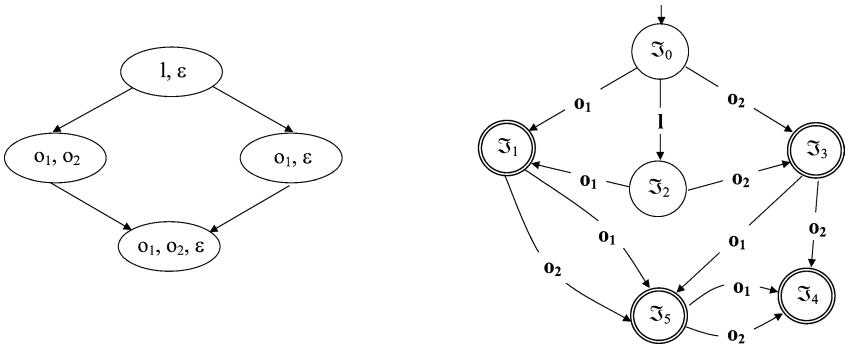

Fig. 5. Temporal observation $\mathcal{O}$ of ξ (left) and relevant index space (right).

(2) If $N \prec N'$ and $N' \prec N''$ then $N \prec N''$;
(3) If $N \mapsto N' \in \mathbf{E}$ then $\nexists N'' \in \mathbf{N}\,(N \prec N'' \prec N')$.

In such a graph, nodes represent uncertain observable events, giving rise to *logical uncertainty*. Under logical uncertainty, an observable event is the disjunction of several labels. A basic assumption is that only one of these labels is the actual observable event generated during the reaction. In particular, when the disjunction includes ε, it may be the case that no observable event was generated. Besides, the partial ordering among nodes supports *temporal uncertainty*. A degenerate case of a temporal observation is the *empty observation*, whose graph does not include any node. The observation $\mathcal{O}$ relevant to a blind diagnostic problem is *null*, $\mathcal{O} = \perp$.[5]

Intuitively, a temporal observation $\mathcal{O}$ is the relaxation of the sequence of observable events that we would expect from the reaction of the system. We know that such a reaction is a sequence of component transitions, some of which (the visible ones) generate an observable label. Thus, if both the content and the absolute ordering of such observable labels were preserved, the system observation should be received as a sequence *Obs* of labels. However, in real, large-scale application domains, noise and multiple communication channels (connecting the system with the observer) are bound to alter both the absolute ordering and the content of each label of *Obs*. On the one hand, noise makes the observable label uncertain, resulting in a set of possible (candidate) labels. On the other, multiple communication channels cause a relaxation of the absolute ordering of *Obs* into a partial ordering. In the end, what is observed is $\mathcal{O}$ instead of *Obs*.

Example 5. Depicted on the left of Fig. 5 is the graph relevant to a temporal observation for the system ξ of Fig. 3, as perceived by the viewer $\mathcal{V}$ defined in Example 3. The top-level node is marked by the set $\{l, \varepsilon\}$, meaning that either l or nothing was first generated by the reaction. The next actually generated observable event is one of the labels within the successive nodes, that is, $\{o_1, o_2\}$ and $\{o_1, \varepsilon\}$. At any rate, before generating the last

[5] An empty observation is different from a null one: in the former nothing is to be seen (even if the viewer is not blind), while in the latter nothing can be seen because of the viewer blindness.

observable event, both these nodes are to be considered since they precede the last node, marked by $\{o_1, o_2, \varepsilon\}$.

Index space

Since it is neither trivial nor efficient to reason about the observation graph as is, an additional DAG is generated, called the *index space* of the temporal observation $\mathcal{O}$, namely $\mathit{Isp}(\mathcal{O})$. The peculiarity of an index space is that each path from the root to a final node, called a *temporal sequence*, represents a mode in which labels may be chosen in the observation graph without violating the constraints imposed by temporal and logical uncertainty [20]. As such, the language $\mathit{Lang}(\mathit{Isp}(\mathcal{O}))$ is the sound and complete set of temporal sequences relevant to the observation graph.

Example 6. Depicted on the right of Fig. 5 is the index space of the observation on the left. Each node is marked by $\Im_i$, $i \in [0\,..\,5]$, where $\Im_0$ is the root, while $\Im_1$, $\Im_3$, $\Im_4$, and $\Im_5$ are final nodes. Accordingly, each path from the root to a final node is a temporal sequence. For instance, the temporal sequence $\langle o_1 \rangle$ corresponds to choosing ε in the first node of the observation, o_1 in the left node, ε in the right node, and ε in the bottom node.[6] Note how $\langle l \rangle$ is not a temporal sequence, as $\Im_2$ is not final. This is consistent with the observation graph, as choosing l in the root node requires, for completing the observation, at least an additional choice between o_1 and o_2 in the left node. A temporal sequence of four observable events corresponds to a choice where no ε is selected, for instance, $\langle l, o_1, o_2, o_2 \rangle$.[7]

Observation restriction

Let $\mathcal{O}$ be an observation for a system Ψ, and Ψ' a subsystem of Ψ. Roughly, the *restriction* of $\mathcal{O}$ on Ψ', denoted $\mathcal{O}_{\langle \Psi' \rangle}$, is an observation for Ψ' obtained by restricting each node of $\mathcal{O}$ on the labels relevant to components in Ψ'. A formal definition of observation restriction is given in [20].

Example 7. With reference to the observation $\mathcal{O}$ for system ξ displayed in Fig. 5, relevant to Example 5, consider the subsystem ξ' of ξ, which is composed of components b_1 and p only. The restriction of $\mathcal{O}$ on ξ' will replace nodes $\{o_1, o_2\}$ and $\{o_1, o_2, \varepsilon\}$ with two identical nodes $\{o_1, \varepsilon\}$, while keeping the other nodes unchanged, namely $\{l, \varepsilon\}$ and $\{o_1, \varepsilon\}$.

6.3. Ruler

A ruler establishes what transitions are to be considered faulty and the granularity of the diagnosis. Formally, the ruler $\mathcal{R}$ of a diagnostic problem $\wp(\Psi)$ is a partial mapping

[6] This choice generates the string $\langle \varepsilon, o_1, \varepsilon, \varepsilon \rangle$, that equals $\langle o_1 \rangle$, as the null label ε is irrelevant.

[7] One may argue that this sequence is not consistent with the behavior space of ξ (Fig. 4), as breaker b_2 cannot open twice in the same reaction. However, the assumption we made on logical uncertainty, namely that one of the labels within the disjunction was actually generated during the reaction, allows for at least one temporal sequence of observable events that is consistent with the behavior space. On the other hand, further consistent sequences (other than the actual sequence) are possibly generated because of the additional 'spurious' labels (logical uncertainty) and the relaxation of the total ordering (temporal uncertainty).

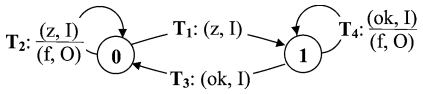

Fig. 6. Extended model of breaker.

between the set $\mathbf{T}$ of component transitions in Ψ and a set of labels $\boldsymbol{\Phi}$ (the faults), $\mathcal{R} : \mathbf{T} \hookrightarrow \boldsymbol{\Phi}$. If $(T, \varphi) \in \mathcal{R}$, then T is *faulty*, otherwise T is *normal*. A label in $\boldsymbol{\Phi}$ that is associated by $\mathcal{R}$ with several transitions in $\mathbf{T}$ is *ambiguous*, and, likewise, $\mathcal{R}$ is ambiguous.

If $\mathcal{R}$ is not ambiguous, $\mathcal{R}$ is a *deep ruler*. An ambiguous ruler such that each fault in $\boldsymbol{\Phi}$ is associated with a subset of the transitions relevant to the same component is a *shallow ruler*. The peculiarity of a shallow ruler lies in the confinement of the ambiguous fault within the scope of a single component.

Example 8. With reference to system ξ of Fig. 3, consider the extension of the breaker model displayed in Fig. 6. The behavior of the breaker is now symmetric, as it involves the additional transitions T_3 and T_4 in state 1 (open), which are triggered by event *ok*, denoting the extinction of the short circuit on the protected device (self-repair). Transition T_3 moves the breaker to state 0 (closed), while T_4, which is the counterpart of T_2, leaves the breaker open. A deep ruler for $\wp(\xi)$ might have the domain of faults $\boldsymbol{\Phi} = \{fo_1, fc_1, fo_2, fc_2\}$, where *fo* and *fc* stand for *fail-to-open* and *fail-to-close*, respectively. The ruler is defined by the following bindings:

$$\mathcal{R} = \big\{\big(T_2(b_1), fo_1\big), \big(T_4(b_1), fc_1\big), \big(T_2(b_2), fo_2\big), \big(T_4(b_2), fc_2\big)\big\}.$$

A shallow ruler $\mathcal{R}'$ might not distinguish between *fail-to-open* and *fail-to-close*, thereby considering the restricted domain of faults $\boldsymbol{\Phi}' = \{f_1, f_2\}$, namely:

$$\mathcal{R}' = \big\{\big(T_2(b_1), f_1\big), \big(T_4(b_1), f_1\big), \big(T_2(b_2), f_2\big), \big(T_4(b_2), f_2\big)\big\}.$$

A diagnosis involving such faults discriminates between breakers, disregarding the actual faulty transitions. However, the ruler is shallow since both f_1 and f_2 are involved in bindings of a single component only, namely b_1 and b_2, respectively. By contrast, ruler $\mathcal{R}''$ defined on domain $\boldsymbol{\Phi}'' = \{fo, fc\}$, with bindings

$$\mathcal{R}'' = \big\{\big(T_2(b_1), fo\big), \big(T_4(b_1), fc\big), \big(T_2(b_2), fo\big), \big(T_4(b_2), fc\big)\big\},$$

is ambiguous but not shallow, as the same fault is associated with transitions of both breakers. A relevant diagnosis discriminates between faults rather than breakers.

6.4. Candidate diagnosis and problem solution

Let $\wp(\Psi) = (\Psi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$ be a diagnostic problem and $h \in \mathit{Lang}(\mathit{Bhv}(\Psi, \Psi_0))$ a relevant history. The *by-product* of h and $\mathcal{V}$ is the sequence

$$h \otimes \mathcal{V} = \big\langle \ell \mid T \in h, (T, \ell) \in \mathcal{V} \big\rangle. \tag{6}$$

Similarly, the by-product of h and $\mathcal{R}$ is the set

$$h \otimes \mathcal{R} = \big\{ \varphi \mid T \in h, (T, \varphi) \in \mathcal{R} \big\}. \tag{7}$$

A *candidate diagnosis* δ of $\wp(\Psi)$ is a set of labels in the domain $\boldsymbol{\Phi}$ of $\mathcal{R}$ such that

$$\delta = h \otimes \mathcal{R}, \quad h \in \mathit{Lang}\big(\mathit{Bhv}(\Psi, \Psi_0)\big), \quad h \otimes \mathcal{V} \in \mathit{Lang}\big(\mathit{Isp}(\mathcal{O})\big). \tag{8}$$

The *solution* of $\wp(\Psi)$, denoted by $\Delta(\wp(\xi))$, is the set of candidate diagnoses of $\wp(\Psi)$.

6.5. Compiled knowledge

Compiled knowledge pertains to a certain initial state of Ψ. We assume that $\mathcal{K}$ is the knowledge relevant to the initial state Ψ_0 expressed in $\wp(\Psi)$. Three classes of compiled-knowledge are envisaged, that are instantiated by relevant *knowledge graphs*, namely:

(1) *Behavior*, whose language is a subset of the language of the behavior space;
(2) *Abduction*, whose language is a subset of the language of the behavior space and where each node is decorated with a relevant candidate diagnosis, in accordance with a ruler;
(3) *Map*, where each path is the sequence of observable events relevant to histories of Ψ, in accordance with a viewer, and each node is marked by a set of candidate diagnoses, in accordance with a ruler.

7. Subsumption

Reusability is a major principle of flexible diagnosis (Requirement 6). Reusing a diagnostic problem $\wp$ for solving a new problem $\wp'$ means exploiting the special-purpose knowledge generated on-line for solving $\wp$ in a way similar to the exploitation of the general-purpose diagnostic knowledge compiled off-line. The aim is to assimilate special-purpose knowledge to general-purpose knowledge when appropriate. Ideally, if this assimilation is fulfilled, the pattern-matching techniques based on general-purpose knowledge are applicable for solving $\wp'$ too.

We need, therefore, to clarify the conditions under which reusability is applicable. To this end, we introduce a relationship called *subsumption*, that is applicable to observations, viewers, and rulers, these being the essential elements of a diagnostic problem. Since subsumption is defined identically for viewers and rulers, we just need to introduce two notions, namely *observation subsumption* and *binding subsumption*, which allow us to eventually define the concept of *problem subsumption*.

7.1. Observation subsumption

Let $\mathcal{O}$ and $\mathcal{O}'$ be two observations.[8] We say that $\mathcal{O}$ *subsumes* $\mathcal{O}'$, written $\mathcal{O} \ni \mathcal{O}'$, when either $\mathcal{O}$ is null or the language of $\mathit{Isp}(\mathcal{O})$ contains the language of $\mathit{Isp}(\mathcal{O}')$:

$$\mathcal{O} \ni \mathcal{O}' \iff \mathcal{O} = \perp \vee \mathit{Lang}\big(\mathit{Isp}(\mathcal{O})\big) \supseteq \mathit{Lang}\big(\mathit{Isp}(\mathcal{O}')\big). \tag{9}$$

Intuitively, assuming the same (non-blind) viewer for both observations, if $\mathcal{O} \ni \mathcal{O}'$ then $\mathcal{O}$ is consistent with all the temporal sequences implicitly specified by the graph of $\mathcal{O}'$, each

[8] No a priori assumption is made on the relationship between $\mathcal{O}$ and $\mathcal{O}'$.

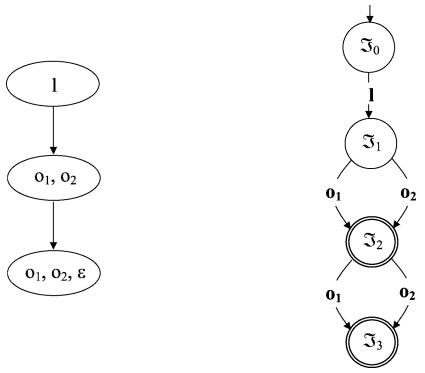

Fig. 7. Temporal observation $\mathcal{O}'$ (left) and relevant index space (right).

element of the space language being one of such sequences. Consequently, the constraints imposed by $\mathcal{O}$ are less stringent than those imposed by $\mathcal{O}'$, as each possible temporal sequence of $\mathcal{O}$ represents a chance of consistency with an additional set of histories for the system and, possibly, an additional set of candidate diagnoses. This is even more evident when $\mathcal{O} = \perp$, as no constraints at all are imposed by $\mathcal{O}$.

Example 9. Shown in Fig. 7 is a temporal observation $\mathcal{O}'$ for system ξ (Fig. 3) that is subsumed by the observation $\mathcal{O}$ displayed in Fig. 5, namely $\mathcal{O} \ni \mathcal{O}'$. Comparing the respective index spaces on the right of the figures, it is easy to check the containment of temporal sequences, $\mathit{Lang}(\mathit{Isp}(\mathcal{O})) \supset \mathit{Lang}(\mathit{Isp}(\mathcal{O}'))$.

7.2. Binding subsumption

The notion of subsumption can be extended to viewers as well as to rulers. Let $\mathbf{T}$ be the whole set of component transitions relevant to a system Ψ, and $\boldsymbol{\Lambda}$ a domain of labels such that $|\boldsymbol{\Lambda}| \leqslant |\mathbf{T}|$. A *binding set* $\mathbf{B}$ is a partial mapping between $\mathbf{T}$ and $\boldsymbol{\Lambda}$, $\mathbf{B}: \mathbf{T} \hookrightarrow \boldsymbol{\Lambda}$, that involves all the labels in $\boldsymbol{\Lambda}$. As such, $\mathbf{B}$ can be expressed as a binary relation between $\mathbf{T}$ and $\boldsymbol{\Lambda}$, with the constraint that at most one label $\ell \in \boldsymbol{\Lambda}$ can be associated with each transition $T \in \mathbf{T}$, namely:

$$\mathbf{B} = \big\{(T_1, \ell_1), (T_2, \ell_2), \ldots, (T_n, \ell_n)\big\} \subseteq \mathbf{T} \times \boldsymbol{\Lambda}. \tag{10}$$

Let ψ and ψ' be two isomorphic systems, and $\mathbf{B}$ and $\mathbf{B}'$ two relevant binding sets, with domains of labels $\boldsymbol{\Lambda}$ and $\boldsymbol{\Lambda}'$, respectively. We say that $\mathbf{B}$ *subsumes* $\mathbf{B}'$, denoted $\mathbf{B} \ni \mathbf{B}'$, iff the following condition holds:

$$\forall (T', \ell') \in \mathbf{B}' \big((T, \ell) \in \mathbf{B},\ T \between T',\ \forall (\mathbb{T}, \ell) \in \mathbf{B}\big((\mathbb{T}', \ell') \in \mathbf{B}',\ \mathbb{T} \between \mathbb{T}'\big)\big). \tag{11}$$

Intuitively, the above formula amounts to the following conditions:

(1) The set of transitions involved in $\mathbf{B}'$ is isomorphic to a subset of the transitions involved in $\mathbf{B}$;

(2) For each label ℓ associated in $\mathbf{B}$ with a transition that has a corresponding transition in $\mathbf{B}'$ associated with ℓ', the set of transitions associated with ℓ in $\mathbf{B}$ is isomorphic to a subset of the transitions associated with ℓ' in $\mathbf{B}'$.

Let ℓ be a label in $\boldsymbol{\Lambda}$. The *renaming* of ℓ within the context of $\mathbf{B}$ and $\mathbf{B}'$ is a symbol in $\boldsymbol{\Lambda}' \cup \{\varepsilon\}$, defined as follows:

$$Ren(\ell, \mathbf{B}, \mathbf{B}') = \begin{cases} \ell' & \text{if } (T, \ell) \in \mathbf{B},\ (T', \ell') \in \mathbf{B}',\ T \between T', \\ \varepsilon & \text{otherwise.} \end{cases} \tag{12}$$

Example 10. With reference to Example 3, it is easy to show that $\mathcal{V} \ni \mathcal{V}'$. Considering Example 8, it is easy to verify that $\mathcal{R} \ni \mathcal{R}'$ and $\mathcal{R} \ni \mathcal{R}''$. By contrast, $\mathcal{R}' \not\ni \mathcal{R}''$, since Formula (10) does not hold with the assignments $T' = T_2(b_1)$, $\ell' = f_1$, $T = T_2(b_1)$, $\ell = fo_1$, $\mathbb{T} = T$, and $\mathbb{T}' = T_2(b_2)$. In fact, $(\mathbb{T}', \ell') = (T_2(b_2), f_1) \notin \mathcal{R}'$. Similarly, it is possible to show that $\mathcal{R}'' \not\ni \mathcal{R}'$.

When $\mathbf{B}$ subsumes $\mathbf{B}'$ and $\mathbf{B}'$ subsumes $\mathbf{B}$, we say $\mathbf{B}$ and $\mathbf{B}'$ are *isomorphic*, namely:

$$\mathbf{B} \doteq \mathbf{B}' \iff (\mathbf{B} \ni \mathbf{B}',\ \mathbf{B}' \ni \mathbf{B}). \tag{13}$$

Proposition 1. *Checking whether there exists a subsumption between two binding sets* $\mathbf{B}$ *and* $\mathbf{B}'$*is in the worst case quadratic.*

7.3. Observation projection

Before introducing the concept of a problem subsumption, we need to define the projection of a temporal observation.[9] Let $\mathcal{O}$ be a temporal observation for a system Ψ and $\mathcal{V}$ a relevant viewer. Let $\mathcal{V}'$ be a different viewer for Ψ, such that $\mathcal{V} \ni \mathcal{V}'$. The projection of a node N in $\mathcal{O}$ on $\mathcal{V}'$, namely $N_{[\mathcal{V}']}$, is a node N' that includes a label ℓ' for each label ℓ in N, where[10]:

$$\ell' = \begin{cases} \varepsilon & \text{if } \ell = \varepsilon, \\ Ren(\ell, \mathcal{V}, \mathcal{V}') & \text{otherwise.} \end{cases} \tag{14}$$

The projection of $\mathcal{O}$ on $\mathcal{V}'$, denoted $\mathcal{O}_{[\mathcal{V}']}$, is an observation $\mathcal{O}'$ where, for each node N in $\mathcal{O}$, $\mathcal{O}'$ includes a node $N' = N_{[\mathcal{V}']}$. The projection of a null observation is null.

Example 11. Consider viewers $\mathcal{V}$ and $\mathcal{V}'$ defined in Example 3. Displayed on top of Fig. 8 are an observation $\mathcal{O}$ (left) relevant to viewer $\mathcal{V}$ and the projection $\mathcal{O}_{[\mathcal{V}']}$ (right). Accordingly, labels in $\mathcal{V}$ have been renamed into corresponding labels in $\mathcal{V}'$, with duplicate removal. For instance, the set of labels $\{o_1, o_2\}$ has been transformed into the singleton $\{o\}$, as $Ren(o_1, \mathcal{V}, \mathcal{V}') = Ren(o_2, \mathcal{V}, \mathcal{V}') = o$. The relevant index spaces are depicted on the bottom (node identifiers are omitted).

[9] Projection should not be confused with the notion of restriction defined in Section 6.2.

[10] The number of labels in N' may possibly decrease owing to duplicate removal.

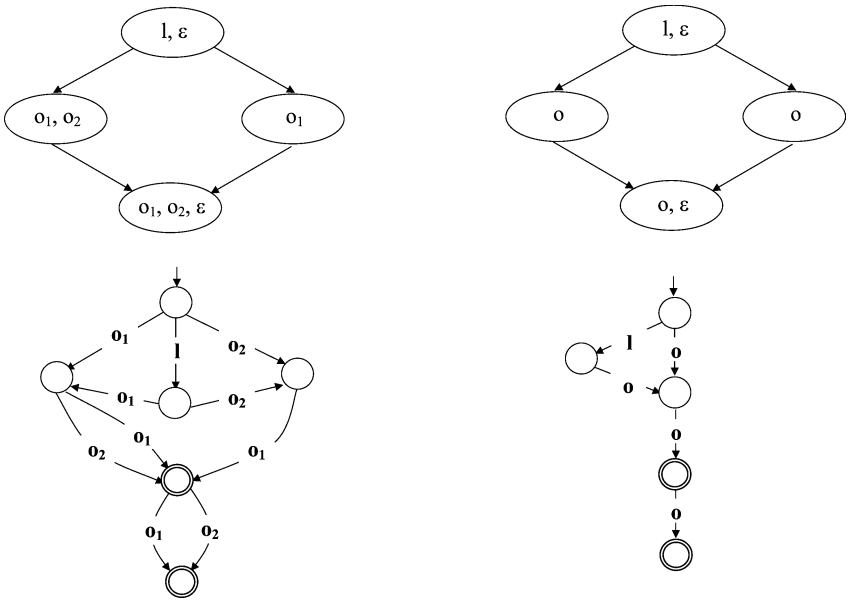

Fig. 8. Observation (left) and relevant projection (right).

Index-space projection

The notion of projection can be extended to index spaces in a natural way. Let $Isp(\mathcal{O})$ be the index space of an observation $\mathcal{O}$ relevant to a viewer $\mathcal{V}$, and $\mathcal{V}'$ a viewer such that $\mathcal{V} \ni \mathcal{V}'$. The projection of $Isp(\mathcal{O})$ on $\mathcal{V}'$, $Isp_{[\mathcal{V}']}(\mathcal{O})$, is the automaton $Isp'(\mathcal{O})$ obtained from $Isp(\mathcal{O})$ as follows:

(1) Replace each label ℓ in $Isp(\mathcal{O})$ with the new label $\ell' = Ren(\ell, \mathcal{V}, \mathcal{V}')$, thereby obtaining (in general) a nondeterministic automaton $Isp^{n}(\mathcal{O})$;
(2) Generate the deterministic automaton $Isp'(\mathcal{O})$ equivalent to $Isp^{n}(\mathcal{O})$.

Example 12. With reference to Fig. 8 (Example 11), it is easy to verify that the projection of the index space of $\mathcal{O}$ (left) on $\mathcal{V}'$ is isomorphic to the index space of $\mathcal{O}_{[\mathcal{V}']}$ (right). This is not incidental, as formalized by the following proposition.

Proposition 2. *Let $\mathcal{O}$ be a temporal observation relevant to a viewer $\mathcal{V}$, and $\mathcal{V}'$ a viewer such that $\mathcal{V} \ni \mathcal{V}'$. Then, the language of the index space of the projection of $\mathcal{O}$ on $\mathcal{V}'$ equals the language of the projection of the index space of $\mathcal{O}$ on $\mathcal{V}'$, namely*

$$Lang\big(Isp(\mathcal{O}_{[\mathcal{V}']})\big) = Lang\big(Isp_{[\mathcal{V}']}(\mathcal{O})\big). \tag{15}$$

Note how such an equivalence does not force *per se* the projection of the index space in order to make up the index space of the projected observation, as it can be generated directly based on the projected observation.

Observation isomorphism

The notions of observation subsumption and projection support the concept of isomorphism between observations. Let $\mathcal{O}$ and $\mathcal{O}'$ be two observations with viewers $\mathcal{V}$ and $\mathcal{V}'$, respectively, such that $\mathcal{V} \doteq \mathcal{V}'$. We say that $\mathcal{O}$ is *isomorphic* to $\mathcal{O}'$ iff the projection of $\mathcal{O}$ on $\mathcal{V}'$ subsumes $\mathcal{O}'$ and the projection of $\mathcal{O}'$ on $\mathcal{V}$ subsumes $\mathcal{O}$:

$$\mathcal{O} \doteq \mathcal{O}' \iff (\mathcal{O}_{[\mathcal{V}']} \ni \mathcal{O}',\ \mathcal{O}'_{[\mathcal{V}]} \ni \mathcal{O}). \tag{16}$$

Intuitively, if $\mathcal{O} \doteq \mathcal{O}'$, the observation graphs of $\mathcal{O}$ and $\mathcal{O}'$ will match in both topology and content of nodes, based on the isomorphism of $\mathcal{V}$ and $\mathcal{V}'$. Such a matching will occur between the relevant index spaces $Isp(\mathcal{O})$ and $Isp(\mathcal{O}')$ too.

7.4. Problem subsumption

Roughly, the solution of a diagnostic problem $\wp'$ may be supported by the knowledge relevant to the solution of a previous diagnostic problem $\wp$, provided that $\wp$ subsume $\wp'$. Intuitively, if such a subsumption holds, the knowledge necessary for solving $\wp'$ is somewhat incorporated within the knowledge generated for solving $\wp$.

Two notions of problem subsumption are defined, namely *weak subsumption* and *strong subsumption*, where the latter is a refinement of (implies) the former.

Let $\wp = (\Psi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$ and $\wp' = (\Psi'_0, \mathcal{V}', \mathcal{O}', \mathcal{R}', \mathcal{K}')$ be two problems for system Ψ and Ψ', respectively, where $\Psi \doteq \Psi'$. We say that $\wp$ *weakly subsumes* $\wp'$, denoted $\wp \sqsupseteq \wp'$, iff $\mathcal{V}$ is subsumed by $\mathcal{V}'$ and $\mathcal{O}$ subsumes the projection of $\mathcal{O}'$ on $\mathcal{V}$:

$$\wp \sqsupseteq \wp' \iff (\mathcal{V} \Subset \mathcal{V}',\ \mathcal{O} \ni \mathcal{O}'_{[\mathcal{V}]}). \tag{17}$$

We say that $\wp$ (*strongly*) *subsumes* $\wp'$, denoted $\wp \ni \wp'$, iff $\wp \sqsupseteq \wp'$ and $\mathcal{R} \ni \mathcal{R}'$:

$$\wp \ni \wp' \iff (\mathcal{V} \Subset \mathcal{V}',\ \mathcal{O} \ni \mathcal{O}'_{[\mathcal{V}]},\ \mathcal{R} \ni \mathcal{R}'). \tag{18}$$

We say that a (strong) problem subsumption is characterized by the *co-variance* of both observation and ruler, and the *contra-variance* of the viewer.

The definition of problem subsumption may sound odd, especially because of the contra-variance property of the viewer. The rationale for the contra-variance of the viewer will be clarified in Section 10. Intuitively, assuming for simplicity $\Psi = \Psi'$, the contra-variance of the viewer allows the observation $\mathcal{O}$ to be *no* more constrained than $\mathcal{O}'$. Consequently, the behavior[11] relevant to $\wp$ will incorporate all the histories of the behavior relevant to $\wp'$. Thus, the behavior of $\wp$ may be reused to solve $\wp'$. Besides, if strong subsumption holds, knowledge reusability may be extended to the graphs involving diagnostic information (abductions and maps).

Example 13. With reference to ξ (Fig. 3, p. 240), consider viewers $\mathcal{V} = \{(T_1(b_1), o), (T_1(b_2), o), (T_2(p), l), (T_3(p), r)\}$, $\mathcal{V}' = \{(T_1(b_1), o_1), (T_1(b_2), o_2), (T_2(p), l), (T_3(p), r)\}$, and rulers $\mathcal{R} = \{(T_2(b_1), f), (T_2(b_2), f)\}$, $\mathcal{R}' = \{(T_2(b_1), f_1), (T_2(b_2), f_2), (T_1(p), s)\}$.

[11] Recall that a behavior is a graph where each path is a possible history of the system (Section 6.5). A behavior consistent with a diagnostic problem $\wp$ is supposed to incorporate the whole set of histories that comply with the problem and, specifically, with the relevant temporal observation.

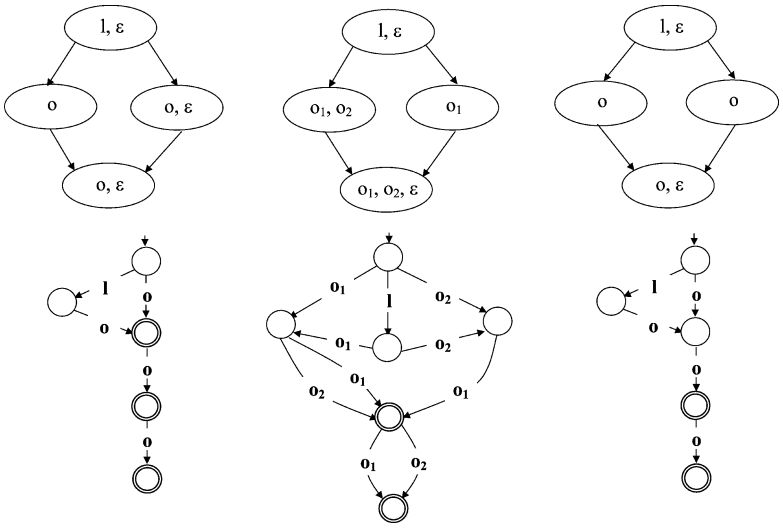

Fig. 9. Observations $\mathcal{O}$, $\mathcal{O}'$, $\mathcal{O}'_{[\mathcal{V}]}$ (top), and corresponding index spaces (bottom).

Also, consider two diagnostic problems for system ξ, namely $\wp = (\xi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$ and $\wp' = (\xi_0, \mathcal{V}', \mathcal{O}', \mathcal{R}', \mathcal{K}')$, where observations $\mathcal{O}$, $\mathcal{O}'$, and $\mathcal{O}'_{[\mathcal{V}]}$ are displayed on the top of Fig. 9, along with relevant index spaces on the bottom. Note the topological similarity between $Isp(\mathcal{O})$ and $Isp(\mathcal{O}'_{[\mathcal{V}]})$, with the discrepancy that the former includes an extra final state. Thus, $Lang(Isp(\mathcal{O})) = \{o, lo, oo, ooo, loo, looo\}$ and $Lang(Isp(\mathcal{O}'_{[\mathcal{V}]})) = \{oo, ooo, loo, looo\}$. Since $Lang(Isp(\mathcal{O})) \supset Lang(Isp(\mathcal{O}'_{[\mathcal{V}]}))$, it follows $\mathcal{O} \ni \mathcal{O}'_{[\mathcal{V}]}$. Besides, since $\mathcal{V} \Subset \mathcal{V}'$ (indeed, this is a precondition for $\mathcal{O}'_{[\mathcal{V}]}$), weak subsumption holds, namely $\wp \sqsupseteq \wp'$. By contrast, since $\mathcal{R} \not\ni \mathcal{R}'$, strong subsumption does not hold, that is, $\wp \not\ni \wp'$.

History isomorphism

When $\wp \sqsupseteq \wp'$, the behavior B generated on-line for solving $\wp$ somehow encompasses all the histories relevant to the behavior B' corresponding to $\wp'$. More accurately, a subset of the histories in B is isomorphic to the set of histories in B'. This leads us to the notion of isomorphism between histories. Let h and h' be two histories within B and B', respectively. We say that $h = \langle T_1, \ldots, T_n \rangle$ and $h' = \langle T'_1, \ldots, T'_n \rangle$ are *isomorphic* iff the following conditions hold:

(1) Both h and h' are rooted in a common initial system state;
(2) Each pair (T_i, T'_i), $i \in [1..n]$, involves corresponding transitions, $T_i \between T'_i$.

Behavior subsumption

We say that a behavior B *subsumes* a behavior B' iff the set of histories in B' is isomorphic to a subset of the histories in B, namely:

$$B \ni B' \iff \big(\forall h' \in Lang(B')\big(h \in Lang(B), h \doteq h'\big)\big). \tag{19}$$

We say that B and B' are *isomorphic* iff they subsume each other, namely:

$$B \doteq B' \iff (B \ni B',\ B' \ni B). \tag{20}$$

Proposition 3. *Let $\wp$ and $\wp'$ be two diagnostic problems such that $\wp$ weakly subsumes $\wp'$. Let B and B' be the behaviors relevant to $\wp$ and $\wp'$, respectively. Then, B subsumes B', namely*

$$\wp \sqsupseteq \wp' \Longrightarrow B \ni B'. \tag{21}$$

Support for Proposition 3 is given in Example 14.

Example 14. With reference to Fig. 9, let B and B' be the behaviors relevant to $\wp$ and $\wp'$, respectively (Example 13). Since $\wp \sqsupseteq \wp'$, we have $B \ni B'$.

Intuitively, the reason why behavior subsumption holds lies in that each path in $Isp(\mathcal{O}')$ is abstracted by a path in $Isp(\mathcal{O})$. For instance, path loo in $Isp(\mathcal{O})$ is an abstraction of the paths lo_1o_1, lo_1o_2, lo_2o_1, and lo_2o_2 in $Isp(\mathcal{O}')$.[12]

Since each path in $Isp(\mathcal{O})$ gives rise to a subset of the behavior (in terms of system histories), the abstraction relationship from paths in $Isp(\mathcal{O}')$ to paths in $Isp(\mathcal{O})$ offers evidence that the part of the behavior relevant to a path in $Isp(\mathcal{O}')$ is included in the part of the behavior relevant to the corresponding (abstract) path in $Isp(\mathcal{O})$. In a nutshell, the more abstract the path, the larger the part of corresponding behavior.

Note that $Isp(\mathcal{O})$ may include paths that do not correspond to any path in $Isp(\mathcal{O}')$. In Fig. 9, one of them is lo (a path is required to end in a final state).

8. Solving model-based problems

In this section, we face the solution of diagnostic problems for which no compiled knowledge is available.[13] To this end we need to formalize the notions of behavior and abduction introduced in Section 6.5.

Behavior

Let $\wp(\Psi) = (\Psi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$ be a diagnostic problem. If $\wp(\Psi)$ is blind ($\mathcal{V} = \emptyset$), then the behavior of $\wp(\Psi)$ coincides with the behavior space $Bhv(\Psi, \Psi_0)$. If $\mathcal{V} \neq \emptyset$, then the behavior of $\wp(\Psi)$ is an automaton $Bhv(\wp(\Psi)) = (\mathbf{S}, \mathbf{E}, \mathbf{T}, \beta_0, \mathbf{S}_f)$, where $\mathbf{S}$ is the set of states $S = (\sigma, \Im)$, such that σ is a state in the behavior space $Bhv(\Psi, \Psi_0)$ and $\Im$ a node of the index space $Isp(\mathcal{O})$, $\mathbf{E}$ is the set of events, that is, a subset of the component transitions

[12] This property is clearly expressed by the subsumption relationship $\mathcal{O} \ni \mathcal{O}'_{[\mathcal{V}]}$, which is reduced to a containment relationship between the corresponding index spaces.

[13] This means that we need reconstructing the system behavior based on the 'crude' system model. Interestingly, the data structures (graphs) yielded for solving a model-based problem are not thrown away, rather they can be possibly exploited in future 'similar' problems, as detailed in Section 10. In particular, the usefulness of the abduction lies on its carrying the diagnostic information based on a specific ruler, which is not the case for behaviors. On the other hand, a behavior can be exploited whichever the ruler of the new diagnostic problem.

in Ψ, $\beta_0 = (\Psi_0, \Im_0)$ is the initial state (where $\Im_0$ is the root of the index space $Isp(\mathcal{O})$), $\mathbf{S}_\mathrm{f}$ is the set of final states,

$$\mathbf{S}_\mathrm{f} = \big\{ S \mid S \in \mathbf{S},\ S = (\sigma, \Im),\ \sigma \text{ is quiescent},\ \Im \text{ is final} \big\}, \tag{22}$$

and $\mathbf{T}$ is the transition function, $\mathbf{T} : \mathbf{S} \times \mathbf{E} \mapsto \mathbf{S}$, where $(\sigma, \Im) \xrightarrow{T} (\sigma', \Im') \in \mathbf{T}$ iff:

(1) $\sigma \xrightarrow{T} \sigma'$ is a transition in $Bhv(\Psi, \Psi_0)$;
(2) If $(T, \ell) \in \mathcal{V}$ then $\Im \xrightarrow{\ell} \Im'$ is in $Isp(\mathcal{O})$ else $\Im' = \Im$.

Furthermore, we require that each state in $S \in \mathbf{S}$ be connected with a state in $\mathbf{S}_\mathrm{f}$, that is, there is a path from S to a final state.

Example 15. Consider the diagnostic problem $\wp(\xi) = (\xi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$ for system ξ, the behavior space $Bhv(\xi, \xi_0)$ of which is depicted in Fig. 4 (p. 241). Assume the temporal observation $\mathcal{O}$ outlined in Fig. 5 (p. 244) and the viewer $\mathcal{V}$ defined in Example 3 (p. 243). The relevant behavior $Bhv(\wp(\xi))$ is depicted in Fig. 10, where dashed arrows and nodes are the inconsistent part of the graph because of the lack of connection with a final node.

Abduction

An abduction $Abd(\wp(\Psi))$ is an automaton $Abd(\wp(\Psi)) = (\mathbf{S}, \mathbf{E}, \mathbf{T}, \alpha_0, \mathbf{S}_\mathrm{f})$, where $\mathbf{S}$ is the set of states $S = (\beta, \delta)$, such that β is a state in the behavior $Bhv(\wp(\Psi))$ and δ is the set of faults, relevant to ruler $\mathcal{R}$, that are pertinent to histories up to S, $\mathbf{E}$ is the same set

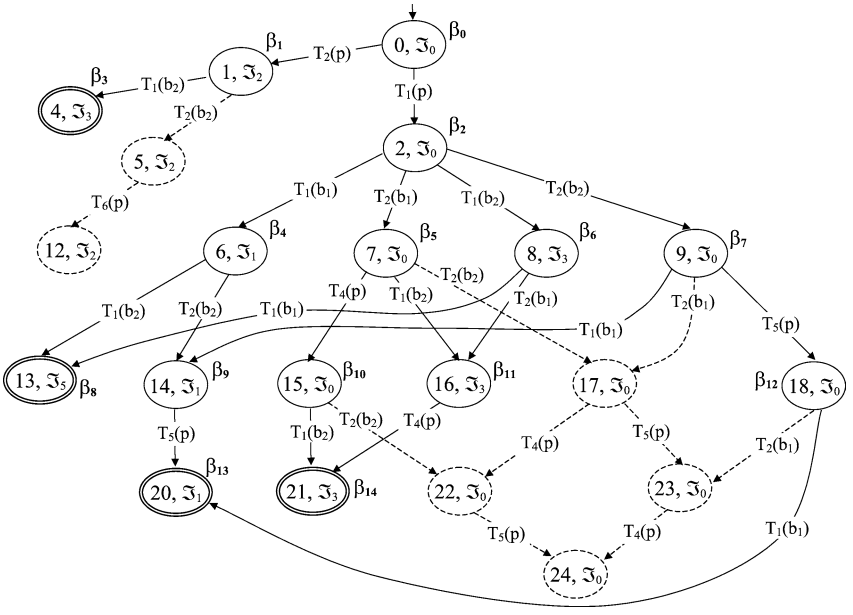

Fig. 10. Behavior $Bhv(\wp(\xi))$.

of events of $Bhv(\wp(\Psi))$, that is, a set of component transitions in Ψ, $\alpha_0 = (\beta_0, \emptyset)$ is the initial state (where β_0 is the initial state of $Bhv(\wp(\Psi))$), $\mathbf{S}_\mathrm{f}$ is the set of final states,

$$\mathbf{S}_\mathrm{f} = \big\{S \mid S \in \mathbf{S},\, S = (\beta, \delta),\, \beta \text{ is a final state in } Bhv\big(\wp(\Psi)\big)\big\}, \tag{23}$$

and $\mathbf{T}$ is the transition function, $\mathbf{T}: \mathbf{S} \times \mathbf{E} \mapsto \mathbf{S}$, where $(\beta, \delta) \xrightarrow{T} (\beta', \delta') \in \mathbf{T}$ iff:

(1) $\beta \xrightarrow{T} \beta'$ is a transition in $Bhv(\wp(\Psi))$;
(2) $\delta' = \delta \cup \{\varphi \mid (T, \varphi) \in \mathcal{R}\}$.

More generally, the notion of an abduction can be applied to a pair $(B, \mathcal{R})$, where B is a behavior and $\mathcal{R}$ a ruler, namely $Abd(B, \mathcal{R})$.

Example 16. Shown in Fig. 11 is the abduction $Abd(\wp(\xi))$ relevant to the behavior $Bhv(\wp(\xi))$ displayed in Fig. 10 (Example 15), assuming ruler $\mathcal{R} = \{(T_1(p), s), (T_2(b_1), f_1), (T_2(b_2), f_2)\}$, where fault s stands for *shorted*, while faults f_1 and f_2 denote *failed-to-open* for breakers b_1 and b_2, respectively. Each node α_i of the abduction is identified by a pair (β_i, δ_i), where β_i is the identifier of a node in $Bhv(\wp(\xi))$, while δ_i is the set of faults yielded by the partial histories up to α_i.

Each set of faults associated with a final state within the abduction is an *abduced diagnosis*. The whole set of abduced diagnoses is the *diagnostic set* of the abduction:

$$\Delta\big(Abd\big(\wp(\Psi)\big)\big) = \big\{\delta \mid (\beta, \delta) \in \mathbf{S}_\mathrm{f}\big\}. \tag{24}$$

Example 17. With reference to Fig. 11, $\Delta(Abd(\wp(\xi)))$ is the set of diagnoses in final nodes α_3, α_8, α_{13}, and α_{14}, that is $\Delta(Abd(\wp(\xi))) = \{\emptyset, \{s\}, \{s, f_1\}, \{s, f_2\}\}$.

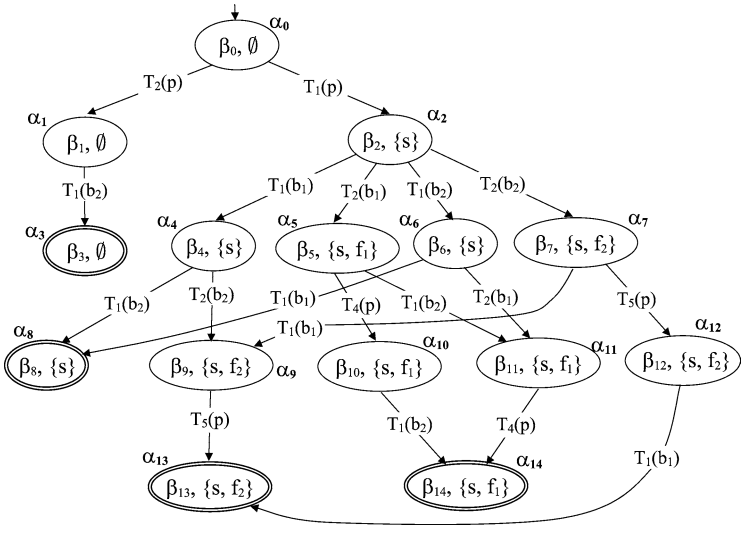

Fig. 11. Abduction $Abd(\wp(\xi))$ relevant to the behavior shown in Fig. 10.

Algorithm 1. The *Abduction* function computes the abduction relevant to a behavior $B = Bhv(\wp(\Psi))$ and a ruler $\mathcal{R}$.[14] At line 1 of the algorithm, the set of states, transitions, and final states of the abduction are initialized. In particular, $\mathbf{S}^{\mathrm{a}}$ is set to a singleton including the initial state $\alpha_0 = (\beta_0, \emptyset)$. The essential part of the algorithm corresponds to the loop within lines 2–11. At each iteration, an unmarked state $\alpha = (\beta, \delta)$ is picked up at line 3 (α_0 is not marked before entering the loop), and all transitions leaving state β in the behavior are considered. Specifically, for each transition, the new set of faults δ' is computed: δ' will differ from δ iff the component transition T is faulty, based on ruler $\mathcal{R}$ (line 5). The sets $\mathbf{T}^{\mathrm{a}}$ and $\mathbf{S}^{\mathrm{a}}$ are updated appropriately at lines 7–8. Note that, when cycles occur, α' might have been yet generated. Once processed all the transitions leaving β in B, the current node α is marked (line 10). When all states in $\mathbf{S}^{\mathrm{a}}$ are marked, in other words, when no new state is generated, the set $\mathbf{S}^{\mathrm{a}}_{\mathrm{f}}$ of final states is made up, at line 12, by selecting those nodes in $\mathbf{S}^{\mathrm{a}}$ that correspond to final states in B.

function *Abduction*$(B, \mathcal{R})$
  input
    $B = (\mathbf{S}, \mathbf{E}, \mathbf{T}, \beta_0, \mathbf{S}_{\mathrm{f}})$: a behavior $Bhv(\wp(\Psi))$,
    $\mathcal{R}$: a ruler for Ψ;
  output
    The abduction $Abd(B, \mathcal{R}) = (\mathbf{S}^{\mathrm{a}}, \mathbf{E}, \mathbf{T}^{\mathrm{a}}, \alpha_0, \mathbf{S}^{\mathrm{a}}_{\mathrm{f}})$;
  begin
1. $\alpha_0 := (\beta_0, \emptyset)$; $\mathbf{S}^{\mathrm{a}} := \{\alpha_0\}$; $\mathbf{T}^{\mathrm{a}} := \emptyset$; $\mathbf{S}^{\mathrm{a}}_{\mathrm{f}} := \emptyset$;
2. **repeat**
3. Get an unmarked state $\alpha = (\beta, \delta)$ in $\mathbf{S}^{\mathrm{a}}$;
4. **for each** transition $\beta \xrightarrow{T} \beta'$ in $\mathbf{T}$ **do**
5. $\delta' := \delta \cup \{\varphi \mid (T, \varphi) \in \mathcal{R}\}$;
6. $\alpha' := (\beta', \delta')$;
7. $\mathbf{T}^{\mathrm{a}} := \mathbf{T}^{\mathrm{a}} \cup \{\alpha \xrightarrow{T} \alpha'\}$;
8. **if** $\alpha' \notin \mathbf{S}^{\mathrm{a}}$ **then** $\mathbf{S}^{\mathrm{a}} := \mathbf{S}^{\mathrm{a}} \cup \{\alpha'\}$
9. **end-for**;
10. Mark α
11. **until** all nodes in $\mathbf{S}^{\mathrm{a}}$ are marked;
12. $\mathbf{S}^{\mathrm{a}}_{\mathrm{f}} := \{\alpha \mid \alpha \in \mathbf{S}^{\mathrm{a}}, \alpha = (\beta, \delta), \beta \in \mathbf{S}_{\mathrm{f}}\}$
  end.

Proposition 4. *$\Delta(Abd(\wp(\Psi)))$ equals the solution of $\wp(\Psi)$.*

A complexity analysis inherent to the model-based method has to take into account the costs of the three involved operations:

[14] The abduction $Abd(\wp(\Psi))$ may be computed directly starting from the model of Ψ, without generating the intermediate behavior $Bhv(\wp(\Psi))$. However, in order to maximize reusability, as clarified in Section 9, it is more convenient generating the abduction based on the behavior.

- The generation of the index space based on the observation, which is exponential in the number of nodes of the observation graph, as proven in [20];
- The generation of the behavior starting from the index space;
- The generation of the abduction starting from the behavior (performed by Algorithm 1).

Proposition 5. *The worst case of the behavior generation starting from the index space is exponential in the product $p \cdot D$, where p is the* (*maximum*) *length of a silent path of the system, given the current viewer, and D is the depth of the index space.*

Proposition 6. *The worst case of the abduction generation starting from the behavior is exponential in the cardinality of the set of faulty transitions defining the ruler.*

9. Knowledge compilation

This section focuses on off-line diagnostic-knowledge compilation. As anticipated in Section 6.5, three classes of knowledge are considered: behaviors, abductions, and maps. As outlined in Section 8, the first two classes are also involved in the solution of model-based problems. However, the diagnostic knowledge generated on-line is special-purpose in nature, since it is tailored to the solution of specific diagnostic problems. We now aim to define knowledge compilation in more general-purpose terms, where knowledge graphs may be exploited by a broad range of diagnostic problems. Let Ψ be a system. A basic assumption of knowledge compilation is that each knowledge graph γ (whether a behavior, an abduction, or a map) is relevant to a subsystem ψ of Ψ, with given initial state ψ_0. Furthermore, γ is not constrained by any specific temporal observation.

The most general knowledge graph is the behavior space $\mathit{Bhv}(\psi, \psi_0)$, as defined in Section 5 (see Example 2, p. 241). In fact, $\mathit{Bhv}(\psi, \psi_0)$ is independent of any viewer or ruler. On the other hand, this generality does not allow for a direct solution of a given diagnostic problem $\wp(\psi) = (\psi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$. In order to determine the solution of $\wp(\psi)$, we need to make up the behavior $\mathit{Bhv}(\wp(\psi))$ constrained by observation $\mathcal{O}$ and viewer $\mathcal{V}$ and, then, the abduction $\mathit{Abd}(\wp(\psi))$ inherent to ruler $\mathcal{R}$.

A more constrained knowledge graph is an *abduction space*, that is, an abduction relevant to a behavior space $\mathit{Bhv}(\psi, \psi_0)$ and a ruler $\mathcal{R}$, namely $\mathit{Abd}(\psi, \psi_0, \mathcal{R})$. Several abduction spaces may be defined for the same behavior space, specifically one for each different ruler. The construction of an abduction space can be carried out by means of Algorithm 1 (p. 256), where the first argument of the *Abduction* function is the behavior space $\mathit{Bhv}(\psi, \psi_0)$. As such, $\mathit{Abd}(\psi, \psi_0, \mathcal{R})$ incorporates all possible histories of system ψ rooted in ψ_0, where each node is decorated with the relevant diagnosis.

Example 18. Shown in Fig. 12 is the abduction space $\mathit{Abd}(\xi, \xi_0, \mathcal{R})$ relevant to the behavior space of Fig. 4 (p. 241) and the ruler $\mathcal{R}$ defined in Example 16.

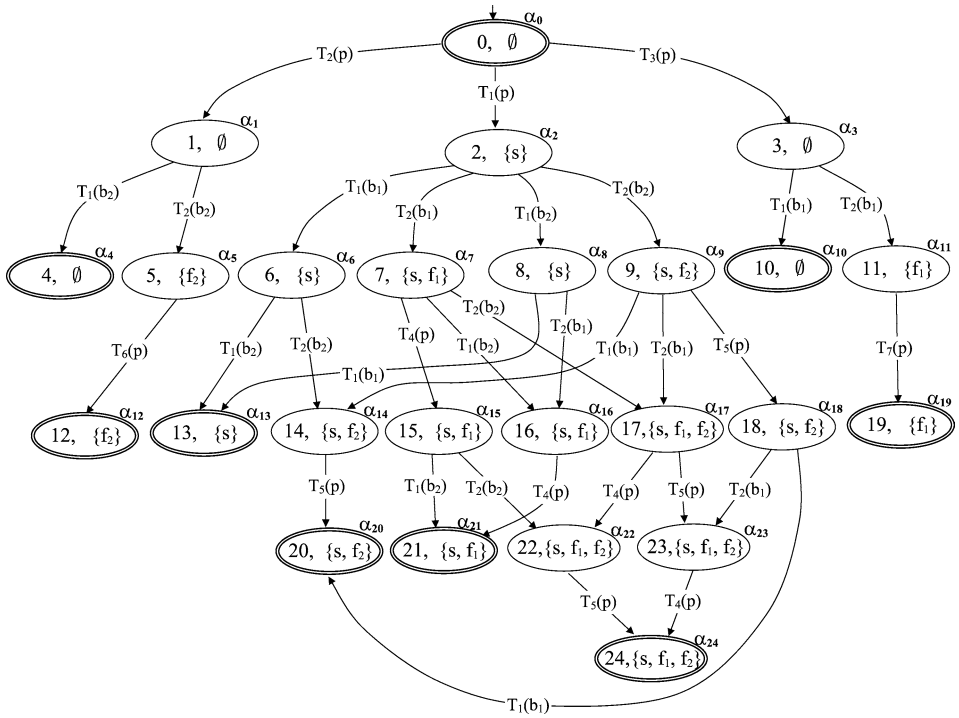

Fig. 12. Abduction space $Abd(\xi, \xi_0, \mathcal{R})$.

The abduction-space independence of any viewer makes it a valuable diagnostic knowledge for a virtually large set of diagnostic problems, specifically, for problems of the kind

$$\wp(\psi) = (\psi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K}),$$

where ψ_0 and $\mathcal{R}$ are matched by $Abd(\psi, \psi_0, \mathcal{R})$. However, such an independence is paid in terms of potential inefficiency in the solution of $\wp(\psi)$. Roughly, solving $\wp(\psi)$ requires generating a restriction of $Abd(\psi, \psi_0, \mathcal{R})$ based on the given observation $\mathcal{O}$ and viewer $\mathcal{V}$, and, then, collecting the abduced diagnoses.

Example 19. Assume the diagnostic problem

$$\wp(\xi) = (\xi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$$

defined in Example 15 (p. 254), where $\mathcal{K}$ encompasses the abduction space $Abd(\xi, \xi_0, \mathcal{R})$ outlined in Fig. 12. To solve $\wp(\xi)$, based on $\mathcal{O}$ (Fig. 5, p. 244) and $\mathcal{V}$ (defined in Example 3, p. 243), we virtually need generating the abduction displayed in Fig. 11 (p. 255) as the restriction of $Abd(\xi, \xi_0, \mathcal{R})$ (Fig. 12) that is constrained by $\mathcal{O}$ and $\mathcal{V}$. The collection of the relevant candidate diagnoses in Fig. 11 determines the solution $\Delta(\wp(\xi))$ outlined in Example 17 (p. 255).

The drawback of using an abduction space $Abd(\xi, \xi_0, \mathcal{R})$ for solving a problem $\wp(\xi) = (\xi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$ lies in the generic nature of the former, specifically, its independence

of viewers. Such an independence prevents the graph from being focused on observable events, since these need a specific viewer $\mathcal{V}$. If $\mathcal{V}$ were known, the abduction space might be simplified so as to reduce the solution of a diagnostic problem to a pattern-matching task, as expected by Requirement 4. This consideration leads us to the notion of a map space. Let

$$\mathit{Abd}(\psi, \psi_0, \mathcal{R}) = (\mathbf{S}, \mathbf{E}, \mathbf{T}, \alpha_0, \mathbf{S}_\mathrm{f}) \tag{25}$$

be an abduction space relevant to a system ψ and a ruler $\mathcal{R}$. Let $\mathcal{V}$ be a viewer for ψ. Let $\mathbf{D}$ be the set of diagnoses associated with the nodes in $\mathbf{S}$. Let

$$\mathit{Map}^\mathrm{n}(\psi, \psi_0, \mathcal{V}, \mathcal{R}) = \big(\mathbf{S}, \mathbf{E}^\mathrm{n}, \mathbf{T}^\mathrm{n}, \alpha_0, \mathbf{S}_\mathrm{f}\big) \tag{26}$$

be the nondeterministic automaton obtained from $\mathit{Abd}(\psi, \psi_0, \mathcal{R})$ by replacing the label T marking each transition $S \xrightarrow{T} S' \in \mathbf{T}$ with the new label ω defined as follows:

$$\omega = \begin{cases} \ell & \text{if } (T, \ell) \in \mathcal{V}, \\ \varepsilon & \text{otherwise.} \end{cases} \tag{27}$$

Let

$$\mathit{Map}^\mathrm{d}(\psi, \psi_0, \mathcal{V}, \mathcal{R}) = \big(\mathbb{S}^\mathrm{d}, \Omega, \mathbb{T}^\mathrm{d}, \mu_0^\mathrm{d}, \mathbb{S}_\mathrm{f}^\mathrm{d}\big) \tag{28}$$

be the deterministic automaton equivalent to $\mathit{Map}^\mathrm{n}(\psi, \psi_0, \mathcal{V}, \mathcal{R})$, where $\mathbb{S}^\mathrm{d} \subset 2^{\mathbf{S}}$ is the set of states,[15] Ω the set of observable events, $\mathbb{T}^\mathrm{d} : \mathbb{S} \times \Omega \mapsto \mathbb{S}$ the transition function, μ_0^d the initial state, and $\mathbb{S}_\mathrm{f}^\mathrm{d} \subseteq \mathbb{S}^\mathrm{d}$ the set of final states. The *map space*, relevant to ψ, ψ_0, $\mathcal{V}$ and $\mathcal{R}$, is the automaton

$$\mathit{Map}(\psi, \psi_0, \mathcal{V}, \mathcal{R}) = (\mathbb{S}, \Omega, \mathbb{T}, \mu_0, \mathbb{S}_\mathrm{f}), \tag{29}$$

isomorphic to $\mathit{Map}^\mathrm{d}(\psi, \psi_0, \mathcal{V}, \mathcal{R})$, where the set of states $\mathbb{S} \subseteq \mathbb{S}^\mathrm{d} \times 2^{\mathbf{D}}$ is such that

$$\forall S \in \mathbb{S}\,\big(S = (S^\mathrm{d}, \mathbb{D}),\ \mathbb{D} = \big\{\delta \mid \alpha = (\beta, \delta), \alpha \in \big(S^\mathrm{d} \cap \mathbf{S}_\mathrm{f}\big)\big\}\big). \tag{30}$$

In other words, the *diagnostic attribute* $\mathbb{D}$ is the set of candidate diagnoses δ associated with the final states α that compose state S^d.

Example 20. Consider the abduction space $\mathit{Abd}(\xi, \xi_0, \mathcal{R})$ displayed in Fig. 12. Assume for system ξ the viewer $\mathcal{V}$ defined in Example 3 (p. 243). The corresponding graph $\mathit{Map}^\mathrm{n}(\xi, \xi_0, \mathcal{V}, \mathcal{R})$ is displayed on the left of Fig. 13, where unlabeled edges are implicitly marked by the null label ε. The relevant map space $\mathit{Map}(\xi, \xi_0, \mathcal{V}, \mathcal{R})$ is outlined on the right of the same figure,[16] whose states, marked by $\mu_0 \ldots \mu_7$, are detailed in Table 1.

[15] The domain of $\mathbb{S}^\mathrm{d} \subset 2^{\mathbf{S}}$ is in accordance with the *subset construction algorithm* [21] which generates the equivalent deterministic automaton, where each state is identified by a subset of the states in the nondeterministic automaton.

[16] Incidentally, all states in $\mathit{Map}(\xi, \xi_0, \mathcal{V}, \mathcal{R})$ are final, since, as detailed in Table 1 and Fig. 12, each state involves at least one final state in $\mathit{Abd}(\xi, \xi_0, \mathcal{R})$.

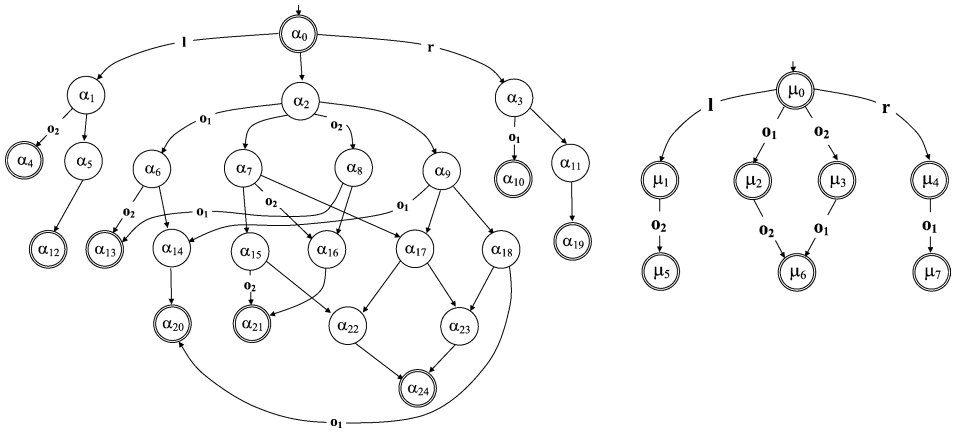

Fig. 13. $Map^{\mathrm{n}}(\xi, \xi_0, \mathcal{V}, \mathcal{R})$ (left) and $Map(\xi, \xi_0, \mathcal{V}, \mathcal{R})$ (right).

Table 1
Node details relevant to $Map(\xi, \xi_0, \mathcal{V}, \mathcal{R})$ in Fig. 13

State	Subset of states in $Abd(\xi, \xi_0, \mathcal{V}, \mathcal{R})$	$\mathbb{D}$
μ_0	$\{\alpha_0, \alpha_2, \alpha_7, \alpha_9, \alpha_{15}, \alpha_{17}, \alpha_{18}, \alpha_{22}, \alpha_{23}, \alpha_{24}\}$	$\{\emptyset, \{s, f_1, f_2\}\}$
μ_1	$\{\alpha_1, \alpha_5, \alpha_{12}\}$	$\{\{f_2\}\}$
μ_2	$\{\alpha_6, \alpha_{14}, \alpha_{20}\}$	$\{\{s, f_2\}\}$
μ_3	$\{\alpha_8, \alpha_{16}, \alpha_{21}\}$	$\{\{s, f_1\}\}$
μ_4	$\{\alpha_3, \alpha_{11}, \alpha_{19}\}$	$\{\{f_1\}\}$
μ_5	$\{\alpha_4\}$	$\{\emptyset\}$
μ_6	$\{\alpha_{13}\}$	$\{\{s\}\}$
μ_7	$\{\alpha_{10}\}$	$\{\emptyset\}$

10. Solving similarity-based problems

Unlike model-based problems coped with in Section 8, similarity-based problems are supported by compiled knowledge, which can be either general-purpose (as outlined in Section 9) or special-purpose (resulting from the solution of a previous problem). We assume that, off-line, after the solution of the model-based problem, the relevant map is possibly generated, based on the abduction and the specific viewer.[17] This allows for the uniform exploitation of compiled knowledge, as, in any case, all three sorts of knowledge graphs are possibly exploitable.[18]

[17] The generation of the (special-purpose) map relevant to the solution of a diagnostic problem is not necessarily computationally complex, even if the system is large: much depends on the degree of constraints imposed by the observation, which can possibly restrict the actual behavior (and abduction) to a manageable size.

[18] Unlike behaviors, abductions and maps are pertinent to specific rulers and viewers. This causes a reduction in knowledge exploitation, even when abduction/map spaces are available.

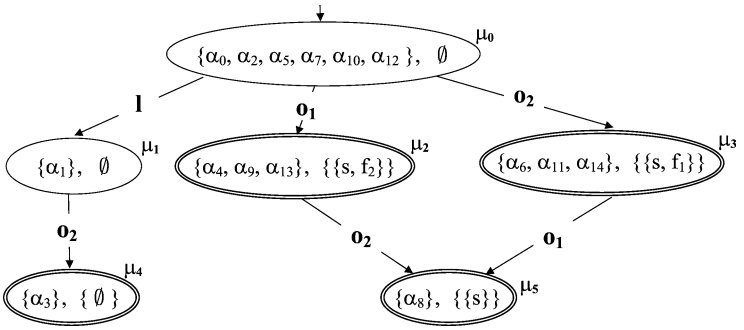

Fig. 14. *Map*($\wp(\xi)$) relevant to *Abd*($\wp(\xi)$) of Fig. 11.

Example 21. Consider the abduction *Abd*($\wp(\xi)$) displayed in Fig. 11 (p. 255). The corresponding *Map*($\wp(\xi)$) is outlined in Fig. 14, where each node is identified by a subset of the states in *Abd*($\wp(\xi)$) and the corresponding set $\mathbb{D}$ of diagnoses. Being relevant to a specific problem, only under subsumption conditions can *Map*($\wp(\xi)$) be exploited for other problems, as shown in Section 10.1.

Compiled problems are classified according to the compiled knowledge. Three classes of problems are defined, namely *μ-problems*, *α-problems*, and *β-problems*, whose solution is possibly supported by maps, abductions, and behaviors, respectively.

Let $\hat{\wp}(\hat{\psi}) = (\hat{\psi}_0, \hat{\mathcal{V}}, \hat{\mathcal{O}}, \hat{\mathcal{R}}, \hat{\mathcal{K}})$ be an actual (non-blind) diagnostic problem for $\hat{\psi}$ to be solved on-line. If $\hat{\wp}(\hat{\psi})$ can be solved by exploiting a map (in $\hat{\mathcal{K}}$) relevant to a system ψ isomorphic to $\hat{\psi}$, then $\hat{\wp}(\hat{\psi})$ is a μ-problem. This is the most efficient way to solve the problem. However, such a suitable map might be not available.[19] In this case, if $\hat{\wp}(\hat{\psi})$ can be solved by exploiting an abduction in $\hat{\mathcal{K}}$, then $\hat{\wp}(\hat{\psi})$ is an α-problem. Solving problems based on abductions is, generally speaking, less efficient than using maps. If neither maps nor abductions are available, chances are that $\hat{\wp}(\hat{\psi})$ be supported by a behavior in $\hat{\mathcal{K}}$. If so, $\hat{\wp}(\hat{\psi})$ is a β-problem. Since no diagnostic information is stored in behaviors, solving β-problems is in general less efficient than solving α-problems or μ-problems.

Before defining the techniques that solve these classes of compiled problems, we need to define the notion of projections applied to diagnoses, map spaces, and histories. Let $\delta = \{\varphi_1, \ldots, \varphi_n\}$ be a set of faults relevant to a ruler $\mathcal{R}$. Let $\mathcal{R}'$ be a ruler such that $\mathcal{R} \ni \mathcal{R}'$. The *projection* of δ on $\mathcal{R}'$ is defined as follows:

$$\delta_{[\mathcal{R}']} = \left\{\varphi' \mid \varphi \in \delta, \varphi' = \mathit{Ren}(\varphi, \mathcal{R}, \mathcal{R}')\right\}. \tag{31}$$

Let $\mathbb{D}$ be a set of diagnoses relevant to $\mathcal{R}$. The projection of $\mathbb{D}$ on $\mathcal{R}'$ is

$$\mathbb{D}_{[\mathcal{R}']} = \left\{\delta' \mid \delta \in \mathbb{D}, \delta' = \delta_{[\mathcal{R}']}\right\}. \tag{32}$$

[19] One may argue that the generation of a behavior may be completed with the generation of a relevant abduction and a relevant map. So, why do not we exploit such a map? The answer is that both abductions and maps are bound to specific rulers and viewers, which are not necessarily the same as (or at least compatible with) those of the current problem.

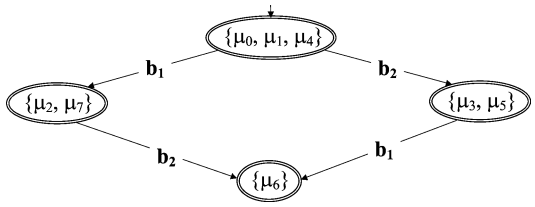

Fig. 15. Projection of the map space $Map(\xi, \xi_0, \mathcal{V}, \mathcal{R})$ shown in Fig. 13.

Example 22. With reference to system ξ outlined in Fig. 3 (p. 240), consider rulers $\mathcal{R} = \{(T_2(b_1), f_1), (T_2(b_2), f_2), (T_1(p), s)\}$ and $\mathcal{R}' = \{(T_2(b_1), f), (T_2(b_2), f)\}$. Let $\mathbb{D} = \{\{s\}, \{s, f_1\}, \{s, f_2\}\}$ be a set of diagnoses relevant to $\mathcal{R}$. Since $\mathcal{R} \ni \mathcal{R}'$, according to our definition, the projection of $\mathbb{D}$ on $\mathcal{R}'$ will be $\mathbb{D}_{[\mathcal{R}']} = \{\emptyset, \{f\}\}$.

Proposition 7. *Let* $\wp(\psi) = (\psi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$, $\wp'(\psi) = (\psi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}', \mathcal{K}')$, *where* $\mathcal{R} \ni \mathcal{R}'$. *Then,*

$$\big(\Delta\big(\wp(\psi)\big)\big)_{[\mathcal{R}']} = \Delta\big(\wp'(\psi)\big). \tag{33}$$

Another relevant concept is that of a map-space projection on a viewer. Let

$$\mathcal{M} = Map(\psi, \psi_0, \mathcal{V}, \mathcal{R}) = (\mathbb{S}, \Omega, \mathbb{T}, \mu_0, \mathbb{S}_f) \tag{34}$$

be a map space for system ψ, and $\mathcal{V}'$ a viewer for ψ', where $\psi \doteq \psi'$ and $\mathcal{V} \ni \mathcal{V}'$. Let $\mathcal{M}^n$ be the nondeterministic automaton obtained from $\mathcal{M}$ by replacing the label ℓ marking each transition $S \xrightarrow{\ell} S' \in \mathbb{T}$ with the new label $\ell' = Ren(\ell, \mathcal{V}, \mathcal{V}')$. The *projection* of $\mathcal{M}$ on $\mathcal{V}'$, $\mathcal{M}_{[\mathcal{V}']}$, is the deterministic automaton equivalent to $\mathcal{M}^n$.

Example 23. Consider the map space $Map(\xi, \xi_0, \mathcal{V}, \mathcal{R})$ on the bottom of Fig. 13 (p. 260), where $\mathcal{V} = \{(T_1(b_1), o_1), (T_1(b_2), o_2), (T_2(p), l), (T_3(p), r)\}$. Let $\mathcal{V}' = \{(T_1(b_1), b_1), (T_1(b_2), b_2)\}$ be a different viewer for ξ, such that $\mathcal{V} \ni \mathcal{V}'$. The projection of $Map(\xi, \xi_0, \mathcal{V}, \mathcal{R})$ on $\mathcal{V}'$ is shown in Fig. 15.

10.1. Solving μ-problems

Consider an actual diagnostic problem $\hat{\wp}(\hat{\psi}) = (\hat{\psi}_0, \hat{\mathcal{O}}, \hat{\mathcal{V}}, \hat{\mathcal{R}}, \hat{\mathcal{K}})$. According to the definition, $\hat{\wp}(\hat{\psi})$ is a μ-problem if $\hat{\mathcal{K}}$ involves a map $\mathcal{M}$ (possibly a map space) that can be exploited for solving $\hat{\wp}(\hat{\psi})$. Whichever the nature of $\mathcal{M}$, the solution of $\hat{\wp}(\hat{\psi})$ requires a sort of matching of $\mathcal{M}$ with the actual (non-null) observation $\hat{\mathcal{O}}$. Formally, assuming that $\mathcal{M}$ and $\hat{\mathcal{O}}$ are relevant to the same viewer, the *matching* of $\mathcal{M}$ with $\hat{\mathcal{O}}$, denoted $\mathcal{M} \asymp \hat{\mathcal{O}}$, is a subgraph of $\mathcal{M}$ that includes all and only the paths of $\mathcal{M}$ which belong to the language of the index space of $\hat{\mathcal{O}}$, that is, such that:

$$Lang(\mathcal{M} \asymp \hat{\mathcal{O}}) = Lang(\mathcal{M}) \cap Lang\big(Isp(\hat{\mathcal{O}})\big). \tag{35}$$

Note how $\mathcal{M} \asymp \hat{\mathcal{O}}$ is still a map for ψ.[20] Similarly to an abduction, we define the *diagnostic set* of a map $\mathcal{M}$ as the union of all the diagnostic attributes associated with the final states of $\mathcal{M}$, namely:

$$\Delta(\mathcal{M}) = \bigcup_{(S^{\mathrm{d}},\mathbb{D}) \in \mathbb{S}_{\mathrm{f}}(\mathcal{M})} \mathbb{D}. \tag{36}$$

The solution of $\hat{\wp}(\hat{\psi}) = (\hat{\psi}_0, \hat{\mathcal{V}}, \hat{\mathcal{O}}, \hat{\mathcal{R}}, \hat{\mathcal{K}})$ is based on Propositions 8 and 9, in which we assume that $\hat{\mathcal{K}}$ incorporates an exploitable map $\mathcal{M}$.

Lemma 1. *If* $\mathcal{M} = \mathit{Map}(\psi, \hat{\psi}_0, \mathcal{V}, \hat{\mathcal{R}})$, $\psi \doteq \hat{\psi}$, *and* $\mathcal{V} \ni \hat{\mathcal{V}}$, *then*

$$\Delta\big(\hat{\wp}(\hat{\psi})\big) = \Delta(\mathcal{M}_{[\hat{\mathcal{V}}]} \asymp \hat{\mathcal{O}}). \tag{37}$$

Lemma 2. *If* $\mathcal{M} = \mathit{Map}(\wp(\psi))$, $\wp(\psi) = (\hat{\psi}_0, \hat{\mathcal{V}}, \mathcal{O}, \hat{\mathcal{R}}, \mathcal{K})$, $\psi \doteq \hat{\psi}$, *and* $\mathcal{O} \ni \hat{\mathcal{O}}$, *then*

$$\Delta\big(\hat{\wp}(\hat{\psi})\big) = \Delta(\mathcal{M} \asymp \hat{\mathcal{O}}). \tag{38}$$

Proposition 8. *If* $\mathcal{M} = \mathit{Map}(\psi, \hat{\psi}_0, \mathcal{V}, \mathcal{R})$, $\psi \doteq \hat{\psi}$, $\mathcal{V} \ni \hat{\mathcal{V}}$, *and* $\mathcal{R} \ni \hat{\mathcal{R}}$, *then*

$$\Delta\big(\hat{\wp}(\hat{\psi})\big) = \big(\Delta(\mathcal{M}_{[\hat{\mathcal{V}}]} \asymp \hat{\mathcal{O}})\big)_{[\hat{\mathcal{R}}]}. \tag{39}$$

Proposition 9. *If* $\mathcal{M} = \mathit{Map}(\wp(\psi))$, $\wp(\psi) = (\hat{\psi}_0, \hat{\mathcal{V}}, \mathcal{O}, \mathcal{R}, \mathcal{K})$, $\psi \doteq \hat{\psi}$, $\mathcal{O} \ni \hat{\mathcal{O}}$, *and* $\mathcal{R} \ni \hat{\mathcal{R}}$, *then*

$$\Delta\big(\hat{\wp}(\hat{\psi})\big) = \big(\Delta(\mathcal{M} \asymp \hat{\mathcal{O}})\big)_{[\hat{\mathcal{R}}]}. \tag{40}$$

Algorithm 2. The *Matching* function implements the matching between a map $\mathcal{M}$ and an observation $\mathcal{O}$. The index space $\mathit{Isp}(\mathcal{O})$ is generated at line 1 (see [20]). The core of the algorithm is the loop within lines 4–15. At each iteration, an unmarked state $(\mu, \Im)$ is picked up at line 5 ($\breve{\mu}_0$ is not marked before entering the loop) and all transitions leaving state μ in $\mathcal{M}$ are considered for matching with the index space: the matching holds when the label ℓ of the transition equals the label of an edge leaving the current state $\Im$ in $\mathit{Isp}(\mathcal{O})$. If so (line 7), the new state $\breve{\mu}'$ is generated (line 8), and the set of states, transitions, and observable events of $\mathcal{M} \asymp \mathcal{O}$ are updated (lines 9–11). Then, the processed state $\breve{\mu}$ is marked. The loop terminates when no further state is to be processed (line 15). The computation of the final states and the possible pruning of states and transitions is performed in lines 16–18.

function *Matching*$(\mathcal{M}, \mathcal{O})$
  input
    $\mathcal{M} = (\mathbb{S}, \Omega, \mathbb{T}, \mu_0, \mathbb{S}_{\mathrm{f}})$: a map for Ψ, relevant to a viewer $\mathcal{V}$,
    $\mathcal{O}$: a temporal observation for Ψ, relevant to $\mathcal{V}$;
  output

[20] More precisely, $\mathcal{M} \asymp \hat{\mathcal{O}}$ carries the same information as a map, in particular, the identifiers of the states in $\mathcal{M}$ and, therefore, the corresponding diagnostic information.

The matching $\mathcal{M} \asymp \mathcal{O} = (\breve{\mathbb{S}}, \breve{\Omega}, \breve{\mathbb{T}}, \breve{\mu}_0, \breve{\mathbb{S}}_f)$;
begin
1. $Isp(\mathcal{O})$:= the index space of $\mathcal{O}$, rooted in $\Im_0$;
2. $\breve{\mu}_0 := (\mu_0, \Im_0)$;
3. $\breve{\mathbb{S}} := \{\breve{\mu}_0\}$; $\breve{\Omega} := \emptyset$; $\breve{\mathbb{T}} := \emptyset$; $\breve{\mathbb{S}}_f := \emptyset$;
4. **repeat**
5. Get an unmarked state $\breve{\mu} = (\mu, \Im)$ in $\breve{\mathbb{S}}$;
6. **for each** transition $\mu \xrightarrow{\ell} \mu'$ in $\mathbb{T}$ **do**
7. **if** $\Im \xrightarrow{\ell} \Im' \in Isp(\mathcal{O})$ **then**
8. $\breve{\mu}' := (\mu', \Im')$;
9. $\breve{\mathbb{T}} := \breve{\mathbb{T}} \cup \{\breve{\mu} \xrightarrow{\ell} \breve{\mu}'\}$;
10. **if** $\breve{\mu}' \notin \breve{\mathbb{S}}$ **then** $\breve{\mathbb{S}} := \breve{\mathbb{S}} \cup \{\breve{\mu}'\}$;
11. **if** $\ell \notin \breve{\Omega}$ **then** $\breve{\Omega} := \breve{\Omega} \cup \{\ell\}$
12. **end-if**
13. **end-for**;
14. Mark $\breve{\mu}$;
15. **until** all nodes in $\breve{\mathbb{S}}$ are marked;
16. $\breve{\mathbb{S}}_f := \{\breve{\mu} \mid \breve{\mu} \in \breve{\mathbb{S}}, \breve{\mu} = (\mu, \Im), \mu \text{ and } \Im \text{ are final}\}$;
17. Remove from $\breve{\mathbb{S}}$ the states that are not connected with a final state;
18. Remove from $\breve{\mathbb{T}}$ the transitions between states that are not in $\breve{\mathbb{S}}$

end.

Proposition 10. *The time complexity of Algorithm* 2 *has an upper bound that is exponential in the number of nodes of the observation graph.*

Example 24. Consider the diagnostic problem $\wp(\xi) = (\xi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$ introduced in Example 15 (p. 254) and continued in Examples 16 and 17, where $\mathcal{O}$ is displayed in Fig. 5 (p. 244), $\mathcal{V} = \{(T_1(b_1), o_1), (T_1(b_2), o_2), (T_2(p), l), (T_3(p), r)\}$, and $\mathcal{R} = \{(T_1(p), s), (T_2(b_1), f_1), (T_2(b_2), f_2)\}$. A relevant map $Map(\xi, \xi_0, \mathcal{V}, \mathcal{R})$ is outlined on the right of Fig. 13 (p. 260). According to Lemma 1, the solution of $\wp(\xi)$ will be $\Delta(\wp(\xi)) = \Delta(Map(\xi, \xi_0, \mathcal{V}, \mathcal{R}) \asymp \mathcal{O})$, where the matching of the map with $\mathcal{O}$ is displayed in Fig. 16. Based on Eq. (36), the candidate set will be the union of the set of diagnoses associated with states μ_2, μ_3, μ_5, and μ_6 (see Table 1, p. 260), namely $\Delta(\wp(\xi)) = \{\emptyset, \{s\}, \{s, f_1\}, \{s, f_2\}\}$, which is in fact the solution found in Example 17 (p. 255), via model-based problem-solving.

Example 25. Considering Example 24, assume a variant $\hat{\wp}(\xi) = (\xi_0, \mathcal{V}, \hat{\mathcal{O}}, \mathcal{R}, \hat{\mathcal{K}})$ of the diagnostic problem $\wp(\xi)$, where $\hat{\mathcal{O}}$ is displayed in Fig. 17, along with the relevant index space. Assume that $\hat{\mathcal{K}}$ embody the map displayed in Fig. 14 (p. 261), which was obtained from the abduction of Fig. 11 (p. 255) (see Example 15, p. 254, and Example 16, p. 255). In particular, such a map is relevant to a diagnostic problem with same viewer and ruler as those in $\hat{\wp}(\xi)$, and the observation outlined in Fig. 5 (p. 244). A comparison of the index spaces of the observations $\mathcal{O}$ in Fig. 5 and $\hat{\mathcal{O}}$ in Fig. 17 clearly shows that $\mathcal{O} \ni \hat{\mathcal{O}}$.

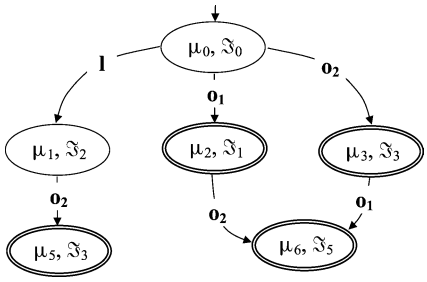

Fig. 16. Matching relevant to Example 24.

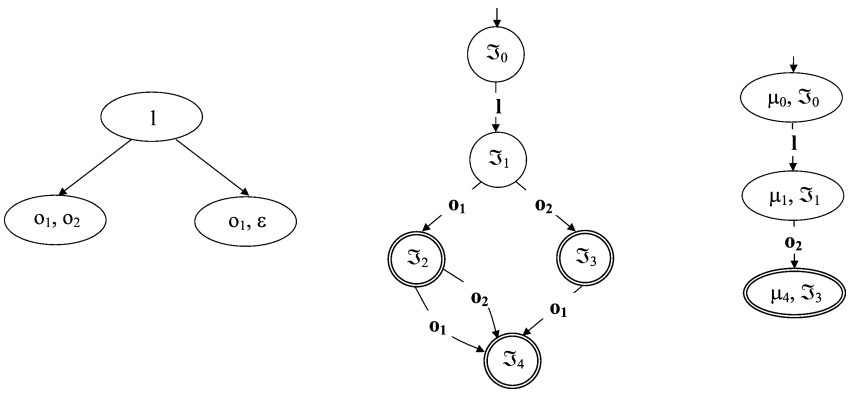

Fig. 17. Observation $\hat{\mathcal{O}}$ (left), relevant index space (center), and matching (right).

Therefore, based on Lemma 2, the solution of $\hat{\wp}(\xi)$ is obtained by extracting the candidates of the matching of the map in Fig. 14 with $\hat{\mathcal{O}}$. Such a matching, which is shown on the right-hand side of Fig. 17, involves the single candidate $\emptyset$ (associated with final state μ_4), that is, $\Delta(\hat{\wp}(\xi)) = \{\emptyset\}$.

10.2. Solving α-problems

A diagnostic problem $\hat{\wp}(\hat{\psi}) = (\hat{\psi}_0, \hat{\mathcal{V}}, \hat{\mathcal{O}}, \hat{\mathcal{R}}, \hat{\mathcal{K}})$ is an α-problem if it is not a μ-problem and $\hat{\mathcal{K}}$ involves an abduction A (possibly an abduction space) that can be exploited for solving $\hat{\wp}(\hat{\psi})$. As for maps, whichever the nature of A, the solution of $\hat{\wp}(\hat{\psi})$ requires a sort of matching of A with the actual (non-null) observation $\hat{\mathcal{O}}$, based on viewer $\hat{\mathcal{V}}$. Formally, the *matching* of A with $\hat{\mathcal{O}}$ based on $\hat{\mathcal{V}}$, denoted $A \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})$, is a subgraph of A that includes all and only the histories of A whose by-product with $\hat{\mathcal{V}}$ belongs to the language of the index space of $\hat{\mathcal{O}}$, that is, such that:

$$Lang\big(A \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})\big) = \big\{h \mid h \in Lang(A),\ h \otimes \hat{\mathcal{V}} \in Lang(Isp(\hat{\mathcal{O}}))\big\}. \tag{41}$$

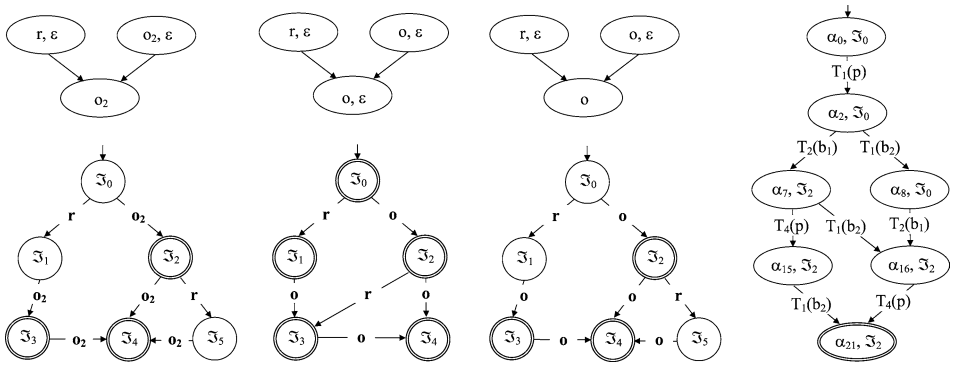

Fig. 18. From left to right, observations $\hat{\mathcal{O}}$, $\mathcal{O}$, $\hat{\mathcal{O}}_{[\mathcal{V}]}$, and matching $A \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})$.

Note how $A \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})$ is still an abduction for ψ.[21] The solution of $\hat{\wp}(\hat{\psi}) = (\hat{\psi}_0, \hat{\mathcal{V}}, \hat{\mathcal{O}}, \hat{\mathcal{R}}, \hat{\mathcal{K}})$ is based on Proposition 11, assuming $A \in \hat{\mathcal{K}}$.

Lemma 3. *If* $A = Abd(\wp(\psi))$, $\wp(\psi) = (\hat{\psi}_0, \mathcal{V}, \mathcal{O}, \hat{\mathcal{R}}, \mathcal{K})$, $\psi \doteq \hat{\psi}$, $\mathcal{V} \Subset \hat{\mathcal{V}}$, *and* $\mathcal{O} \Supseteq \hat{\mathcal{O}}_{[\mathcal{V}]}$, *then*

$$\Delta\big(\hat{\wp}(\hat{\psi})\big) = \Delta\big(A \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})\big). \tag{42}$$

Proposition 11. *If* $A = Abd(\wp(\psi))$, $\wp(\psi) = (\hat{\psi}_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$, $\psi \doteq \hat{\psi}$, $\mathcal{V} \Subset \hat{\mathcal{V}}$, $\mathcal{O} \Supseteq \hat{\mathcal{O}}_{[\mathcal{V}]}$, *and* $\mathcal{R} \Supseteq \hat{\mathcal{R}}$, *then*

$$\Delta\big(\hat{\wp}(\hat{\psi})\big) = \big(\Delta\big(A \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})\big)\big)_{[\hat{\mathcal{R}}]}. \tag{43}$$

Example 26. Consider a diagnostic problem $\hat{\wp}(\xi) = (\xi_0, \hat{\mathcal{V}}, \hat{\mathcal{O}}, \hat{\mathcal{R}}, \hat{\mathcal{K}})$, where $\hat{\mathcal{O}}$ and $Isp(\hat{\mathcal{O}})$ are displayed on the left-hand side of Fig. 18, $\hat{\mathcal{V}} = \{(T_1(b_1), o_1), (T_1(b_2), o_2), (T_2(p), l), (T_3(p), r)\}$, and $\hat{\mathcal{R}} = \{(T_1(p), s), (T_2(b_1), f_1), (T_2(b_2), f_2)\}$. Assume that $\hat{\mathcal{K}}$ encompass the abduction A relevant to a (previously) solved diagnostic problem $\wp(\xi) = (\xi_0, \mathcal{V}, \mathcal{O}, \hat{\mathcal{R}}, \mathcal{K})$, where $\mathcal{O}$ and $Isp(\mathcal{O})$ are outlined in the second column of Fig. 18, and $\mathcal{V} = \{(T_1(b_1), o), (T_1(b_2), o), (T_2(p), l), (T_3(p), r)\}$. Note that $\mathcal{V} \Subset \hat{\mathcal{V}}$. Furthermore, the projection $\hat{\mathcal{O}}_{[\mathcal{V}]}$ depicted in the third column of Fig. 18 offers evidence for the relationship $\mathcal{O} \Supseteq \hat{\mathcal{O}}_{[\mathcal{V}]}$. Thus, based on Lemma 3, the solution of $\hat{\wp}(\xi)$ can be generated by distilling the candidates from the matching $A \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})$. As to A, it is easy to verify that it corresponds to the subgraph of the abduction displayed in Fig. 12 (p. 258), which is obtained by removing the subtree leaving state α_0 through transition $T_2(p)$ (that is, by removing states α_1, α_4, α_5, and α_{12}). The matching of A with $\hat{\mathcal{O}}$ based on $\hat{\mathcal{V}}$ is shown on the right-hand side of Fig. 18. Being α_{21} the abduction state involved in the final node, we conclude that the solution of $\hat{\wp}(\xi)$ is the singleton $\{\{s, f_1\}\}$. This result is corroborated by matching the

[21] More precisely, $A \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})$ carries the same information as an abduction, in particular, the identifiers of the states in A and, therefore, the corresponding diagnostic information.

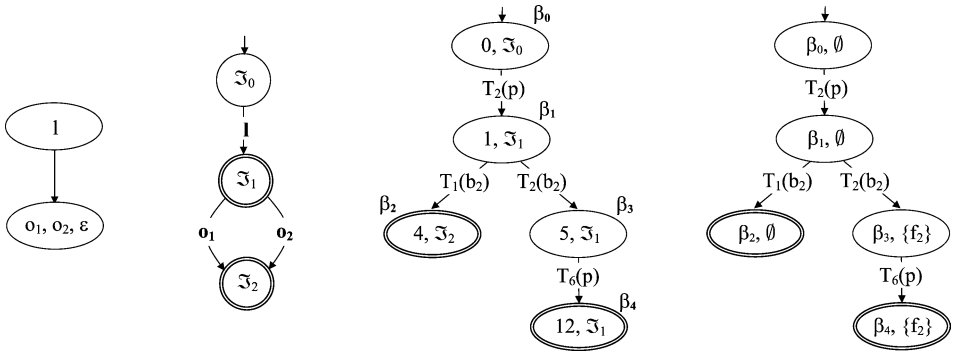

Fig. 19. From left to right, $\hat{\mathcal{O}}$, $Isp(\hat{\mathcal{O}})$, $\hat{B} = (Bhv(\xi, \xi_0) \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}}))$, and $Abd(\hat{B}, \hat{\mathcal{R}})$.

map of Fig. 13 (p. 260) with $\hat{\mathcal{O}}$, whose final node is $(\mu_3, \Im_2)$ (see Table 1, p. 260, for the diagnostic attribute associated with μ_3).

10.3. Solving β-problems

A diagnostic problem $\hat{\wp}(\hat{\psi}) = (\hat{\psi}_0, \hat{\mathcal{V}}, \hat{\mathcal{O}}, \hat{\mathcal{R}}, \hat{\mathcal{K}})$ is a β-problem if it is neither a μ-problem nor an α-problem and $\hat{\mathcal{K}}$ involves a behavior B (possibly a behavior space) that can be exploited for solving $\hat{\wp}(\hat{\psi})$. As for maps and abductions, whichever the nature of B, the solution of $\hat{\wp}(\hat{\psi})$ requires a sort of matching of B with the actual (non-null) observation $\hat{\mathcal{O}}$, based on viewer $\hat{\mathcal{V}}$. Formally, the *matching* of B with $\hat{\mathcal{O}}$, denoted $B \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})$, is a subgraph of B that includes all and only the histories of B whose by-product with $\hat{\mathcal{V}}$ belongs to the language of the index space of $\hat{\mathcal{O}}$, that is, such that:

$$Lang\big(B \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})\big) = \big\{h \mid h \in Lang(B), h \otimes \hat{\mathcal{V}} \in Lang\big(Isp(\hat{\mathcal{O}})\big)\big\}. \tag{44}$$

Note how $B \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})$ is still a behavior for ψ.[22] The solution of $\hat{\wp}(\hat{\psi}) = (\hat{\psi}_0, \hat{\mathcal{V}}, \hat{\mathcal{O}}, \hat{\mathcal{R}}, \hat{\mathcal{K}})$ is based on Proposition 12, assuming $B \in \hat{\mathcal{K}}$.[23]

Proposition 12. *If* $B = Bhv(\wp(\psi))$, $\wp(\psi) = (\hat{\psi}_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$, $\psi \doteq \hat{\psi}$, $\mathcal{V} \Subset \hat{\mathcal{V}}$, *and* $\mathcal{O} \ni \hat{\mathcal{O}}_{[\mathcal{V}]}$, *then*

$$\Delta\big(\hat{\wp}(\hat{\psi})\big) = \Delta\big(Abd\big(\big(B \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})\big), \hat{\mathcal{R}}\big)\big). \tag{45}$$

Example 27. Consider $\hat{\wp}(\xi) = (\xi_0, \hat{\mathcal{V}}, \hat{\mathcal{O}}, \hat{\mathcal{R}}, \hat{\mathcal{K}})$, where $\hat{\mathcal{O}}$ and $Isp(\hat{\mathcal{O}})$ are displayed on the left-hand side of Fig. 19, $\hat{\mathcal{V}} = \{(T_1(b_1), o_1), (T_1(b_2), o_2), (T_2(p), l), (T_3(p), r)\}$, and $\hat{\mathcal{R}} = \{(T_1(p), s), (T_2(b_1), f_1), (T_2(b_2), f_2)\}$. Assume that $\hat{\mathcal{K}}$ includes the behavior space

[22] More precisely, $B \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})$ carries the same information as a behavior, in particular, the identifiers of the states in B.

[23] The fact that solving a β-problem requires the generation of a relevant abduction does not mean that the β-problem is transformed into an α-problem, as the α-problem requires the availability of a *given* exploitable abduction in $\hat{\mathcal{K}}$, which is not the case for the β-problem. On the other hand, the abduction generated for solving the β-problem can possibly become the exploitable knowledge on which a *subsequent* α-problem is based.

$Bhv(\xi, \xi_0)$ displayed in Fig. 4 (p. 241). The solution of $\hat{\wp}(\xi)$ can be generated based on Proposition 12 ($\wp(\psi)$ is blind). Specifically, $\hat{B} = (Bhv(\xi, \xi_0) \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}}))$ and $Abd(\hat{B}, \hat{\mathcal{R}})$ are computed as shown in the right-hand side of Fig. 19. According to the latter, the solution of $\hat{\wp}(\xi)$ is $\{\emptyset, \{f_2\}\}$. The same result is obtained by matching the map space of Fig. 13 (p. 260) with $\hat{\mathcal{O}}$, involving the final nodes μ_1 and μ_5.

11. Fragmented problems

The basic assumption for the solution of the three classes of compiled problems $\hat{\wp}(\hat{\psi})$ defined in Section 10 is the availability of a suitable knowledge graph relevant to another (not necessarily distinct) system ψ such that $\psi \doteq \hat{\psi}$. However, on the one hand, it may be the case that no such a graph be available within the model-based knowledge $\hat{\mathcal{K}}$. On the other, the preprocessing of the global system Ψ might produce partial knowledge in terms of graphs relevant to a partition of ψ. According to Requirement 5, such compiled knowledge is supposed to be exploited in a modular way during on-line diagnosis. This leads us to the concept of a fragmented problem.

A *fragmentation* $\psi^* = \{\psi_1, \ldots, \psi_n\}$ of a system ψ is a set of subsystems of ψ such that $\{\mathbf{C}_1, \ldots, \mathbf{C}_n\}$ is a partition of the components in ψ, where each $\mathbf{C}_i$, $i \in [1\,..\,n]$, is the set of components in ψ_i. A diagnostic problem $\wp(\psi) = (\psi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$ is a *fragmented problem* (or, equivalently, a *ϕ-problem*) iff there exists a fragmentation ψ^* such that:

(1) $\mathcal{K} \supset \{B_1, \ldots, B_n\}$, where $\forall i \in [1\,..\,n]$, B_i is a behavior $Bhv(\wp(\psi_i))$ relevant to a (possibly blind) problem $\wp(\psi_i) = (\psi_{i_0}, \mathcal{V}_i, \mathcal{O}_i, \mathcal{R}_i, \mathcal{K}_i)$, such that:

$$\mathcal{V}_i \Subset \mathcal{V}_{\langle\psi_i\rangle}, \quad \mathcal{O}_i \ni (\mathcal{O}_{\langle\psi_i\rangle})_{[\mathcal{V}_i]}; \tag{46}$$

(2) ψ_0 equals the composition of the initial states ψ_{i_0}, namely, $\psi_0 = (\psi_{1_0}, \ldots, \psi_{n_0})$.

In other words, if $\wp(\psi)$ is a fragmented problem, the knowledge will incorporate a group of behaviors for a fragmentation of ψ, with the subsumption relationships stated in Eq. (46). Intuitively, the behavior $Bhv(\wp(\psi))$ can be generated on-line based on the behaviors relevant to the fragmentation, thereby fulfilling Requirement 5 on modularity.

In the simplest case, $Bhv(\wp(\psi))$ can be generated by joining behaviors $B_1, \ldots, B_n$ within the context of $\wp(\psi)$, where each B_i is thought of as the 'communicating automaton' of ψ_i. That is, each ψ_i is seen as a sort of virtual component with automaton B_i, so that $Bhv(\wp(\psi))$ can be generated based on the B_i, the links among ψ_i, and the constraints imposed by observation $\mathcal{O}$ and viewer $\mathcal{V}$ of $\wp(\psi)$. Intuitively, the *join* of the set of behaviors[24] $B^* = \{B_1, \ldots, B_n\}$ within the context of $\mathcal{V}$ and $\mathcal{O}$, $Join(B^*, \mathcal{V}, \mathcal{O})$, is a behavior of ψ with the same language as $Bhv(\wp(\psi))$:

$$Lang\big(Join(B^*, \mathcal{V}, \mathcal{O})\big) = Lang\big(Bhv\big(\wp(\psi)\big)\big). \tag{47}$$

[24] The *join* operator generates a new behavior as an aggregation of the behaviors in B^*. As such, it is not a binary operator like the join operator of Relational Algebra [22].

More precisely,

$$Join(B^*, \mathcal{V}, \mathcal{O}) = (\mathbf{S}, \mathbf{E}, \mathbf{T}, \beta_0, \mathbf{S}_f) \tag{48}$$

is an automaton defined as follows. $\mathbf{S}$ is the set of states $(\mathbb{B}, \Im, \mathbb{L})$, where $\mathbb{B}$ belongs to the Cartesian product of states in B^*, $\Im$ is a state of $Isp(\mathcal{O})$, and $\mathbb{L}$ is the record of configurations of links among systems in ψ^*. $\mathbf{E}$ is the union of the set of events (component transitions) of behaviors in B^*. $\beta_0 = (\mathbb{B}_0, \Im_0, \mathbb{L}_0)$ is the initial state, where $\mathbb{B}_0$ is the record of initial states of behaviors in B^*, $\Im_0$ is the initial state of $Isp(\mathcal{O})$, and $\mathbb{L}_0$ is the record of empty configurations for links among systems in ψ^*. $\mathbf{S}_f$ is the set of final states $(\mathbb{B}_f, \Im_f, \mathbb{L}_f)$, where all states in $\mathbb{B}_f$ are final in the corresponding behaviors in B^*, $\Im_f$ is final in $Isp(\mathcal{O})$, and all link configurations in $\mathbb{L}_f$ are empty. $\mathbf{T}: \mathbf{S} \times \mathbf{E} \mapsto \mathbf{S}$ is the transition function defined as follows:

$$S \xrightarrow{T} S' \in \mathbf{T} \tag{49}$$

where $S = (\mathbb{B}, \Im, \mathbb{L})$, $S' = (\mathbb{B}', \Im', \mathbb{L}')$, iff the following conditions hold:

(1) T is the label marking a transition exiting a state S_i in $\mathbb{B}$ in a corresponding behavior $B_i \in B^*$;
(2) Either T is silent or the relevant visible label defined in viewer $\mathcal{V}_i$ marks an edge exiting $\Im$ in $Isp(\mathcal{O})$;
(3) The input event triggering T comes either from:
  (a) the standard input *In*,
  (b) a dangling terminal of ψ,
  (c) a configuration in $\mathbb{L}$, or
  (d) a configuration relevant to a link internal to a system in ψ^*;
(4) $\mathbb{B}'$ equals $\mathbb{B}$ apart from the ith element, which is instead the state reached by T in B_i;
(5) If T is silent then $\Im' = \Im$ else $\Im'$ is the state reached by the transition in $Isp(\mathcal{O})$ marked by the observable label of T;
(6) If T is triggered by an event relevant to a link in $\mathbb{L}$ then $\mathbb{L}'$ equals $\mathbb{L}$ minus the triggering event plus the output events of T generated on links relevant to $\mathbb{L}$;
(7) If T is triggered by an event relevant to either the standard input or a dangling terminal or a link internal to a system in ψ^* then $\mathbb{L}'$ equals $\mathbb{L}$ plus the output events of T generated on links relevant to $\mathbb{L}$;[25]
(8) There exists a path from S' to a final state in $\mathbf{S}_f$.

Equivalence (47) provides a formal basis for the fulfillment of Requirement 5, insofar as the partial knowledge corresponding to the behaviors relevant to the fragmentation of the system ψ are exploited by the join operator to make up the behavior of the diagnostic problem $\wp(\psi)$ to be solved on-line. Once yielded $Bhv(\wp(\psi))$, the solution of $\wp(\psi)$ is generated as illustrated in Section 8, by constructing the abduction $Abd(\wp(\psi))$ based on $Bhv(\wp(\psi))$ and collecting the relevant diagnostic set.

One may argue that this is model-based problem-solving, as it follows the steps of behavior and abduction generation. However, this is not the case, as the construction of the

[25] According to Section 4, if the link is full, the output event is lost.

behavior is based on the fragmented knowledge corresponding to the behaviors relevant to the fragmentation, each fragment B_i of knowledge being a sort of compiled model for the relevant subsystem ψ_i.

Proposition 13. *The asymptotic upper bound of the Join operator is exponential in the depth of the resulting behavior graph.*

11.1. Diagnostic tree

Roughly, a diagnostic tree relevant to an actual diagnostic problem $\hat{\wp}(\Psi)$ corresponds to a recursive decomposition of $\hat{\wp}(\Psi)$ into a hierarchy of subproblems relevant to subsystems of Ψ. Each node of the tree is associated with a subsystem $\psi \subseteq \Psi$, along with relevant initial state, observation, and viewer. Essentially, the way the problem is decomposed is only constrained by specific subsumption relationships between viewers and observations, respectively.

Formally, let $\hat{\wp}(\Psi) = (\Psi_0, \hat{\mathcal{O}}, \hat{\mathcal{V}}, \hat{\mathcal{R}}, \hat{\mathcal{K}})$ be a ϕ-problem relevant to a fragmentation

$$\Psi^* = \{\psi_1, \ldots, \psi_n\} \tag{50}$$

and corresponding set of behaviors $B^* = \{B_1, \ldots, B_n\}$. Let $\boldsymbol{\Psi}$ be the domain of subsystems $\psi \subseteq \Psi$ corresponding to the composition of a group of subsystems in Ψ^*. Let $\boldsymbol{\Psi}_0$ be the domain of initial states for systems in $\boldsymbol{\Psi}$. Let $\mathbf{V}$ and $\mathbf{O}$ be the domains of viewers and observations, respectively, for systems in $\boldsymbol{\Psi}$. A *diagnostic tree* relevant to $\hat{\wp}(\Psi)$ and Ψ^* is a 3-tuple

$$\mathit{Dtree}\big(\hat{\wp}(\Psi), \Psi^*\big) = (\mathbf{N}, \mathbf{E}, N_0) \tag{51}$$

where $\mathbf{N} \subseteq \boldsymbol{\Psi} \times \boldsymbol{\Psi}_0 \times \mathbf{V} \times \mathbf{O}$ is the set of *nodes* $N = (\psi, \psi_0, \mathcal{V}, \mathcal{O})$, $\mathbf{E} : \mathbf{N} \mapsto 2^{\mathbf{N}}$ is the set of *edges*, and N_0 is the *root*, such that, denoting with $\mathit{Succ}(N)$ the set of successive nodes of N,

$$\mathit{Succ}(N) : \mathbf{N} \mapsto 2^{\mathbf{N}} = \{N' \mid N \to N' \in \mathbf{E}\}, \tag{52}$$

and with $\mathit{Leaf}(N)$ the Boolean function indicating whether N is a leaf node,

$$\mathit{Leaf}(N) : \mathbf{N} \mapsto \mathit{Boolean} = \begin{cases} \mathit{true} & \text{if } \mathit{Succ}(N) = \emptyset, \\ \mathit{false} & \text{otherwise} \end{cases} \tag{53}$$

the following conditions hold:

(1) $N_0 = (\Psi, \Psi_0, \hat{\mathcal{V}}, \hat{\mathcal{O}})$;
(2) The set of leaves $\{N_1, \ldots, N_n\}$ is isomorphic to Ψ^*, so that:

$$\forall i \in [1 \,..\, n] \quad \big(N_i = (\psi_i, \psi_{i_0}, \mathcal{V}_i, \mathcal{O}_i), \psi_i \in \Psi^*\big); \tag{54}$$

(3) $\forall N_i \in \mathbf{N}$, $\mathit{Leaf}(N_i)$, $N_i = (\psi_i, \psi_{i_0}, \mathcal{V}_i, \mathcal{O}_i)$, we have:

$$\begin{aligned} &B_i = \mathit{Bhv}\big(\wp(\psi_i)\big), \\ &\wp(\psi_i) = (\psi_{i_0}, \mathcal{V}_i, \mathcal{O}_i, \mathcal{R}_i, \mathcal{K}_i), \quad (\wp(\psi_i) \text{ possibly blind}) \end{aligned} \tag{55}$$

where $B_i \in B^*$ is the behavior associated with ψ_i;

(4) $\forall N \in \mathbf{N}$, $\neg Leaf(N)$, $N = (\psi, \psi_0, \mathcal{V}, \mathcal{O})$, where

$$Succ(N) = \big\{N'_j \mid N'_j = (\psi'_j, \psi'_{j_0}, \mathcal{V}'_j, \mathcal{O}'_j),\ j \in [1\,..\,m]\big\}, \tag{56}$$

the following relationships hold:

$$\psi_0 = (\psi'_{1_0}, \ldots, \psi'_{m_0}), \tag{57}$$

$$\forall j \in [1\,..\,m]\ \big(\mathcal{V}'_j \Subset \mathcal{V}_{\langle\psi'_j\rangle}, \mathcal{O}'_j \ni (\mathcal{O}_{\langle\psi'_j\rangle})_{[\mathcal{V}'_j]}\big). \tag{58}$$

We associate with each $N \in \mathbf{N}$, $N = (\psi, \psi_0, \mathcal{V}, \mathcal{O})$, $Succ(N) = \{N'_1, \ldots, N'_m\}$, a behavior by means of the Bhv^{t} recursive function defined as follows:

$$Bhv^{\mathrm{t}}(N) = \begin{cases} B \text{ (the behavior associated with } \psi \text{ in } B^*) & \text{if } Leaf(N), \\ Join(\{Bhv^{\mathrm{t}}(N'_1), \ldots, Bhv^{\mathrm{t}}(N'_m)\}, \mathcal{V}, \mathcal{O}) & \text{otherwise.} \end{cases} \tag{59}$$

Example 28. Shown in Fig. 20 is the system Ξ corresponding to the network of devices depicted in Fig. 1 (p. 234). Assume a diagnostic problem $\wp(\Xi) = (\Xi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$ relevant to a fragmentation $\Xi^* = \{\xi_1, \xi_2, \xi_3, \xi_4\}$, where each ξ_i, $i \in [1\,..\,4]$, is an instantiation of the system ξ portrayed in Fig. 3 (p. 240). The set of behaviors corresponding to Ξ^* is $B^* = \{B_1, \ldots, B_4\}$, each B_i being an instantiation of the behavior space displayed in Fig. 4 (p. 241). The observation $\mathcal{O}$ is outlined on the right (top) of Fig. 21, while $\mathcal{V}$ and $\mathcal{R}$ are defined as follows:

$$\mathcal{V} = \bigcup_{i=1}^{4} \big\{\big(T_1(b_{i1}), o_{i1}\big), \big(T_1(b_{i2}), o_{i2}\big), \big(T_2(p_i), l_i\big), \big(T_3(p_i), r_i\big)\big\},$$

$$\mathcal{R} = \bigcup_{i=1}^{4} \big\{\big(T_1(p_i), s_i\big), \big(T_2(b_{i1}), f_{i1}\big), \big(T_2(b_{i2}), f_{i2}\big)\big\}.$$

A corresponding diagnostic tree is shown in Fig. 22, where ξ_5 and ξ_6 (relevant to nodes N_1 and N_2) are the subsystems obtained by composing ξ_1 and ξ_2, and ξ_3 and ξ_4, respectively. We assume that the initial state of both breakers and protections be 0. According to the terminology introduced in Section 6.1, for each leaf node $N_7 \ldots N_{10}$, the viewer is blind and, consequently, the observation is null. For the remaining nodes $N_0 \ldots N_6$, observations and viewers are specified in Fig. 21 and Table 2, respectively. Unlike $\mathcal{V}_1 \ldots \mathcal{V}_3$,

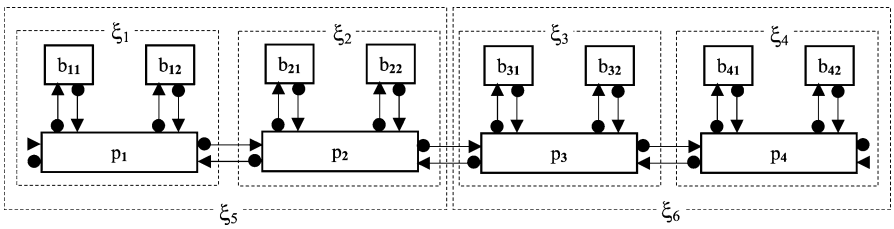

Fig. 20. Device network (top) and corresponding modeled system Ξ (bottom).

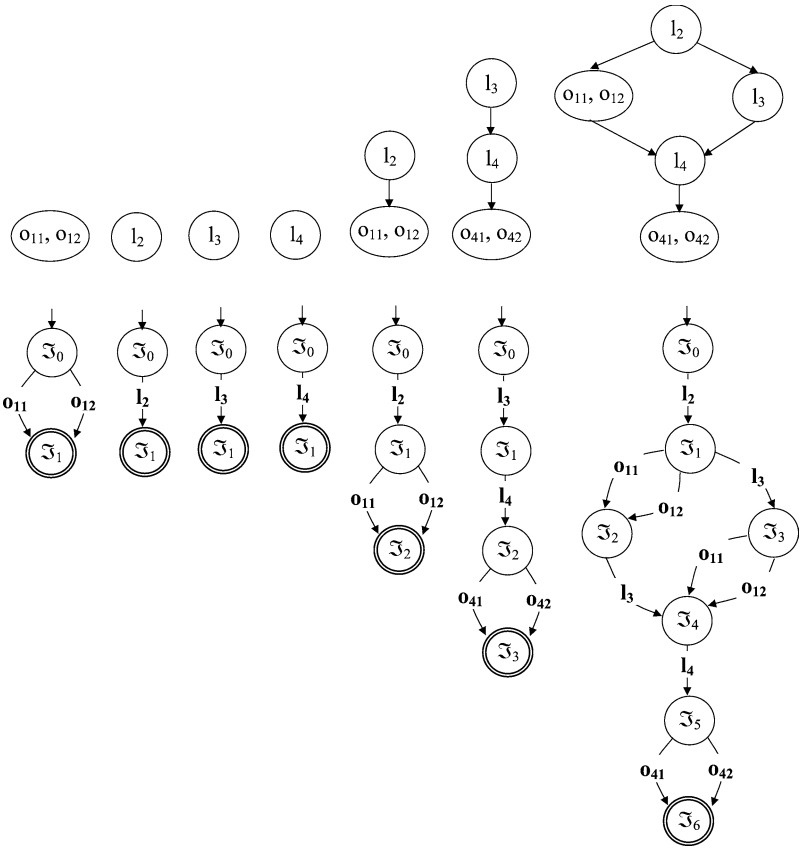

Fig. 21. From left to right, observation graphs (top) and relevant index spaces (bottom) for (sub)systems ξ_1, ξ_2, ξ_3, ξ_4, ξ_5, ξ_6, and Ξ (see Fig. 20), respectively. Corresponding viewers are specified in Table 2.

Table 2
Viewers relevant to the observations displayed in Fig. 21

System	Viewer	Associations
ξ_1	$\mathcal{V}_1$	$\{(T_1(b_{11}), o_{11}), (T_1(b_{12}), o_{12}), (T_2(p_1), l_1), (T_3(p_1), r_1)\}$
ξ_2	$\mathcal{V}_2$	$\{(T_1(b_{21}), o_{21}), (T_1(b_{22}), o_{22}), (T_2(p_2), l_2), (T_3(p_2), r_2)\}$
ξ_3	$\mathcal{V}_3$	$\{(T_1(b_{31}), o_{31}), (T_1(b_{32}), o_{32}), (T_2(p_3), l_3), (T_3(p_3), r_3)\}$
ξ_4	$\mathcal{V}_4$	$\{(T_2(p_4), l_4), (T_3(p_4), r_4)\}$
ξ_5	$\mathcal{V}_5$	$\mathcal{V}_1 \cup \mathcal{V}_2$
ξ_6	$\mathcal{V}_6$	$\mathcal{V}_3 \cup \mathcal{V}_4 \cup \{(T_1(b_{41}), o_{41}), (T_1(b_{42}), o_{42})\}$
Ξ	$\mathcal{V}$	$\mathcal{V}_5 \cup \mathcal{V}_6$

viewer $\mathcal{V}_4$ (relevant to node N_6 in Fig. 22) is not merely the restriction of $\mathcal{V}$ on ξ_4 but a subset of it instead (as no transition of b_{41} or b_{42} is observable in $\mathcal{V}_4$). It is easy to verify that conditions (1–4) for a diagnostic tree are met in Fig. 22. In particular, note how Eq. (58) is true when viewers and observations in child nodes are a restriction of the viewer and

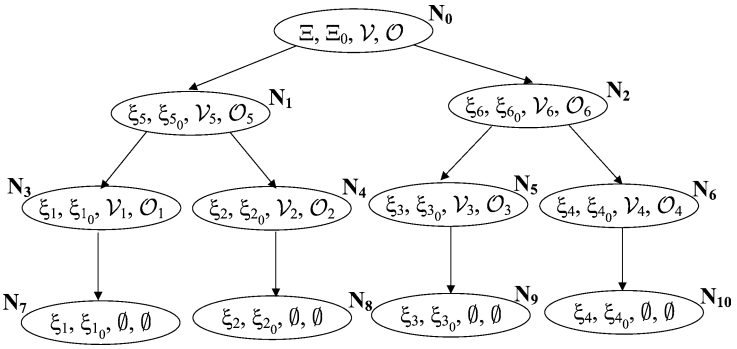

Fig. 22. Diagnostic tree relevant to system Ξ depicted in Fig. 20. Relevant observations and viewers are outlined in Fig. 21 and Table 2.

observation in the parent node, as for instance when the parent node is N_1. An even more trivial case is when a child node N'_j involves a blind viewer $\mathcal{V}'_j$, as the condition in Eq. (58) becomes

$$\emptyset \Subset \mathcal{V}_{\langle \psi'_j \rangle}, \quad \emptyset \Supset \emptyset$$

which is certainly true. This holds in Fig. 22 for parent nodes $N_3 \cdots N_6$. The only non-trivial case involves node N_6 as a child node of N_2. Indeed, $\mathcal{V}_4$ is not the restriction of $\mathcal{V}_6$ on ξ_4 but, rather, a subset of such a restriction, which involves the visible transitions of p_4 only. In this case, the condition in Eq. (58) becomes

$$\mathcal{V}_4 \Subset \mathcal{V}_{6\langle \xi_4 \rangle}, \quad \mathcal{O}_4 \Supset (\mathcal{O}_{\langle \xi_4 \rangle})_{[\mathcal{V}_4]}$$

where $\mathcal{V}_4 \subset \mathcal{V}_{6\langle \xi_4 \rangle}$ and $\mathcal{O}_4 = (\mathcal{O}_{\langle \xi_4 \rangle})_{[\mathcal{V}_4]}$. Thus, the condition is still met.

Proposition 14. *Let $N = (\psi, \psi_0, \mathcal{V}, \mathcal{O})$ be a node of a diagnostic tree and $\wp(\psi) = (\psi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$ a diagnostic problem for ψ. Then,*

$$Lang\big(Bhv^{\mathrm{t}}(N)\big) = Lang\big(Bhv\big(\wp(\psi)\big)\big). \tag{60}$$

Corollary 14.1. *Let $(\mathbf{N}, \mathbf{E}, N_0)$ be a diagnostic tree relevant to $\wp(\Psi)$. Then,*

$$Lang\big(Bhv^{\mathrm{t}}(N_0)\big) = Lang\big(Bhv\big(\wp(\Psi)\big)\big). \tag{61}$$

Corollary 14.2. *Let $(\mathbf{N}, \mathbf{E}, N_0)$ and $(\mathbf{N}', \mathbf{E}', N'_0)$ be two diagnostic trees relevant to the same diagnostic problem $\wp(\Psi)$. Then,*

$$Lang\big(Bhv^{\mathrm{t}}(N_0)\big) = Lang\big(Bhv^{\mathrm{t}}(N'_0)\big) = Lang\big(Bhv\big(\wp(\Psi)\big)\big). \tag{62}$$

Corollary 14.2 represents the formal basis for the preservation of the soundness and completeness of the diagnostic method. In other words, whichever the recursive decomposition of a ϕ-problem, the resulting behavior (associated with the root) is always the same. Consequently, the choice of a particular diagnostic tree is bound to influence the efficiency of the reconstruction process, rather than the result, which, based on Proposition 14, is in fact the behavior relevant to the solution of the diagnostic problem.

The actual generation of $Bhv(\wp(\Psi))$ can be carried out by applying the *Join* operator upward within the diagnostic tree. A node N can be processed only if all its successive nodes have been processed already. In particular, the computation starts from the internal nodes N such that $Succ(N)$ involves leaf nodes only. Eventually, this bottom-up computation leads to the generation of $Bhv(\wp(\Psi))$.

11.2. Diagnostic graph

The naive application of the technique for the generation of the system behavior based on a diagnostic tree is possibly bound to compute the 'same' behavior several times, when nodes are similar. Two nodes of a diagnostic tree, $N = (\psi, \psi_0, \mathcal{O}, \mathcal{V})$ and $N' = (\psi', \psi'_0, \mathcal{O}', \mathcal{V}')$, are *isomorphic* iff the relevant systems, viewers, and observations are isomorphic, respectively, and they share the same initial state:

$$N \doteq N' \iff (\psi \doteq \psi',\ \psi_0 = \psi'_0,\ \mathcal{V} \doteq \mathcal{V}',\ \mathcal{O} \doteq \mathcal{O}'). \tag{63}$$

Proposition 15. *Let N and N' be two nodes of a diagnostic tree such that $N \doteq N'$. Then,*

$$Bhv^{\mathrm{t}}(N) \doteq Bhv^{\mathrm{t}}(N'). \tag{64}$$

Proposition 15 opens the way for processing-reuse when solving fragmented problems. Indeed, based on the diagnostic tree, the bottom-up computation of the behavior relevant to the diagnostic problem may possibly involve the processing of isomorphic nodes. If so, the computation of the behavior is based on a projection operation defined as follows.

Let ψ and ψ' be two systems such that $\psi \doteq \psi'$, and $B = (\mathbf{S}, \mathbf{E}, \mathbf{T}, \beta_0, \mathbf{S}_{\mathrm{f}})$ a behavior relevant to ψ. The *projection* of B on ψ' is a behavior relevant to ψ',

$$B_{[\psi']} = B' = (\mathbf{S}, \mathbf{E}', \mathbf{T}', \beta_0, \mathbf{S}_{\mathrm{f}}), \tag{65}$$

where $|\mathbf{E}'| = |\mathbf{E}|$, $|\mathbf{T}'| = |\mathbf{T}|$, and

$$\forall S_1 \xrightarrow{T} S_2 \in \mathbf{T}\ (S_1 \xrightarrow{T'} S_2 \in \mathbf{T}', T \between T'). \tag{66}$$

Intuitively, B and $B_{[\psi']}$ only differ in the labels marking the edges of the behaviors.

The notions of isomorphism between nodes of a diagnostic tree and of behavior projection allows for a more efficient solution of fragmented problems. Roughly, a graph is generated by a sort of factorization of the diagnostic tree based on the isomorphism of nodes. The actual bottom-up computation of the behavior relevant to the fragmented problem is performed on such a graph rather than on the original tree. The advantage comes from the reduction of the number of nodes on which the computation of the relevant behavior is to be done.

Let $D^{\mathrm{t}} = Dtree(\hat{\wp}(\Psi), \Psi^*) = (\mathbf{N}^{\mathrm{t}}, \mathbf{E}^{\mathrm{t}}, N_0)$ be a diagnostic tree relevant to a diagnostic problem $\hat{\wp}(\Psi)$ and a fragmentation Ψ^*, as defined in Section 11.1. Let $\boldsymbol{\Psi}^{\mathrm{t}}$ be the domain of subsystems of Ψ relevant to the nodes in D^{t}, namely:

$$\boldsymbol{\Psi}^{\mathrm{t}} = \{\psi \mid N \in \mathbf{N}^{\mathrm{t}}, N = (\psi, \psi_0, \mathcal{V}, \mathcal{O})\}. \tag{67}$$

A *diagnostic graph* of D^{t} is a 3-tuple,

$$Dgraph(D^{\mathrm{t}}) = (\mathbf{N}^{\mathrm{g}}, \mathbf{E}^{\mathrm{g}}, N_0), \tag{68}$$

where $\mathbf{N}^{\mathrm{g}} \subseteq \mathbf{N}^{\mathrm{t}}$ is the set of *nodes*, $\mathbf{E}: \mathbf{N}^{\mathrm{g}} \times \boldsymbol{\Psi}^{\mathrm{t}} \mapsto \mathbf{N}^{\mathrm{g}}$ is the set of *edges*, and N_0 is the *root*, such that, denoting with $\aleph^{\mathrm{t}}$ the least partition of $\mathbf{N}^{\mathrm{t}}$ where nodes are grouped by the isomorphism relationship,

$$\aleph^{\mathrm{t}} = \left\{ \mathbb{N} \mid \forall N \in \mathbb{N},\ \forall N' \in \mathbb{N}\ (N \doteq N') \right\}, \tag{69}$$

the following conditions hold:

$$|\mathbf{N}^{\mathrm{g}}| = |\aleph^{\mathrm{t}}|, \tag{70}$$

$$\forall N \in \mathbf{N}^{\mathrm{g}}\ (N \in \mathbb{N},\ \mathbb{N} \in \aleph^{\mathrm{t}}), \tag{71}$$

$$\forall N \xrightarrow{\psi''} N' \in \mathbf{E}^{\mathrm{g}}\ \left(N \to N'' \in \mathbf{E}^{\mathrm{t}},\ N'' = (\psi'', \psi_0'', \mathcal{V}'', \mathcal{O}''),\ N' \doteq N''\right), \tag{72}$$

$$\forall N \to N' \in \mathbf{E}^{\mathrm{t}},\ N \in \mathbf{N}^{\mathrm{g}},\ N' = (\psi', \psi_0', \mathcal{V}', \mathcal{O}')\ (N \xrightarrow{\psi'} N'' \in \mathbf{E}^{\mathrm{g}},\ N'' \doteq N'). \tag{73}$$

As for a diagnostic tree, we associate with each node $N \in \mathbf{N}^{\mathrm{g}}$, $N = (\psi, \psi_0, \mathcal{V}, \mathcal{O})$, a behavior by means of the Bhv^{g} recursive function defined as follows:

$$Bhv^{\mathrm{g}}(N) = \begin{cases} B \text{ (the behavior associated with } \psi \text{ in } B^*) & \text{if } \mathit{Leaf}(N), \\ \mathit{Join}(\mathbb{B}, \mathcal{V}, \mathcal{O}) & \text{otherwise,} \end{cases} \tag{74}$$

where

$$\mathbb{B} = \left\{ B' \mid N \xrightarrow{\psi''} N' \in \mathbf{E}^{\mathrm{g}},\ N' = (\psi', \psi_0', \mathcal{V}', \mathcal{O}'), \right.$$
$$\left. B' = \begin{cases} Bhv^{\mathrm{g}}(N') & \text{if } \psi'' = \psi' \\ (Bhv^{\mathrm{g}}(N'))_{[\psi'']} & \text{otherwise} \end{cases} \right\}. \tag{75}$$

Example 29. Consider the diagnostic tree D^{t} displayed in Fig. 22. A diagnostic graph D^{g} corresponding to D^{t} is shown on the right of Fig. 23. Based on Eq. (69), we have $\aleph = \{\{N_0\}, \{N_1\}, \{N_2\}, \{N_3\}, \{N_4, N_5\}, \{N_6\}, \{N_7, N_8, N_9, N_{10}\}\}$. Such a partition is shown on the left of Fig. 23, which replicates D^{t} in more abstract terms, where elements of $\aleph$ are grouped within dashed boxes. A comparison between D^{t} and D^{g} shows that conditions (70)–(73) are fulfilled. Specifically, conditions (70)–(71) require an isomorphism

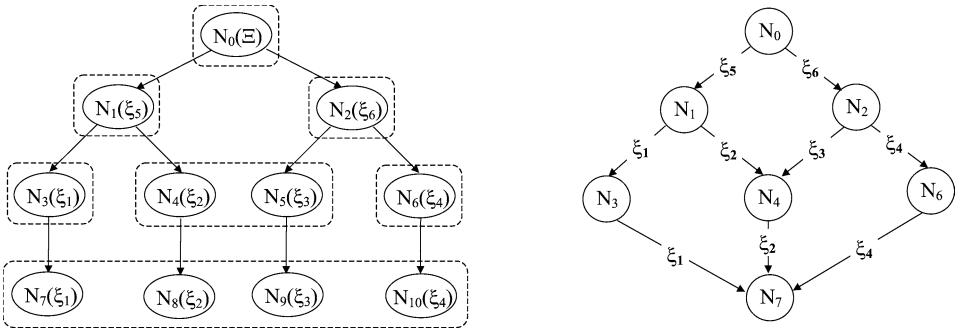

Fig. 23. Diagnostic tree D^{t} (left) and relevant diagnostic graph D^{g} (right).

between $\aleph$ and nodes in D^{g}, where each node in D^{g} is picked up from the elements within $\aleph$. Condition (72) states that for each edge E^{g} in D^{g}, there exists in D^{t} an edge leaving the initial node of E^{g} and entering a node whose system field equals the label marking E^{g}. For example, considering the edge $E^{\mathrm{g}} = N_2 \stackrel{\xi_3}{\rightarrow} N_4$, condition (72) is met by setting $N'' = N_5$. Condition (73) establishes that, for each edge E^{t} in D^{t} leaving a node included in D^{g} too, there exists in D^{g} an edge leaving the same node as E^{t} and marked by the system field relevant to the node entered by E^{t}. For example, for $E^{\mathrm{t}} = N_4 \rightarrow N_8$, Condition (73) is met by $N'' = N_7$.

Proposition 16. *Let D^{t} be a diagnostic tree and $D^{\mathrm{g}} = Dgraph(D^{\mathrm{t}}) = (\mathbf{N}^{\mathrm{g}}, \mathbf{E}^{\mathrm{g}}, N_0)$ the relevant diagnostic graph. Then,*

$$\forall N \in \mathbf{N}^{\mathrm{g}} \left(Lang\big(Bhv^{\mathrm{g}}(N)\big) = Lang\big(Bhv^{\mathrm{t}}(N)\big)\right). \tag{76}$$

Corollary 16.1. *Let $N = (\psi, \psi_0, \mathcal{V}, \mathcal{O})$ be a node of a diagnostic graph and $\wp(\psi) = (\psi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$ a diagnostic problem for ψ. Then,*

$$Lang\big(Bhv^{\mathrm{g}}(N)\big) = Lang\big(Bhv\big(\wp(\psi)\big)\big). \tag{77}$$

Corollary 16.2. *Let $(\mathbf{N}^{\mathrm{g}}, \mathbf{E}^{\mathrm{g}}, N_0)$ be a diagnostic graph relevant to a diagnostic problem $\wp(\Psi)$. Then,*

$$Lang\big(Bhv^{\mathrm{g}}(N_0)\big) = Lang\big(Bhv\big(\wp(\Psi)\big)\big). \tag{78}$$

Corollary 16.3. *Let $(\mathbf{N}^{\mathrm{g}}, \mathbf{E}^{\mathrm{g}}, N_0)$ and $(\mathbf{N}^{\mathrm{g}'}, \mathbf{E}^{\mathrm{g}'}, N_0')$ be two diagnostic graphs relevant to the same diagnostic problem $\wp(\Psi)$. Then,*

$$Lang\big(Bhv^{\mathrm{g}}(N_0)\big) = Lang\big(Bhv^{\mathrm{g}}(N_0')\big) = Lang\big(Bhv\big(\wp(\Psi)\big)\big). \tag{79}$$

Proposition 16 states the equivalence of the functions Bhv^{t} and Bhv^{g}. That is, the application of Bhv^{g} on a node of the diagnostic graph yields a behavior that is equivalent to that generated by applying Bhv^{t} on the same node in the diagnostic tree. This property allows Proposition 14 and Corollaries 14.1–14.2 to hold within the diagnostic graph too, as claimed by Corollaries 16.1–16.3, respectively.

11.3. Solving ϕ-problems

A ϕ-problem $\hat{\wp}(\psi)$ is essentially a β-problem where the compiled knowledge, namely the behavior, is fragmented rather than monolithic. The fragmented knowledge is a set of behaviors relevant to a partition of system ψ such that each behavior fulfills the subsumption relationships outlined in Eq. (46). Thus, the essential trouble is to make up the behavior of ψ based on the these 'behavioral fragments'. Once generated the behavior of ψ consistent with $\hat{\wp}(\psi)$, the solution of $\hat{\wp}(\psi)$ can be found out by first making up the abduction relevant to $Bhv(\hat{\wp}(\psi))$ and, then, by distilling the diagnostic set. The solution of $\hat{\wp}(\psi)$ is based on Proposition 17.

Proposition 17. *Let* $\hat{\wp}(\psi) = (\psi_0, \hat{\mathcal{V}}, \hat{\mathcal{O}}, \hat{\mathcal{R}}, \hat{\mathcal{K}})$ *and* ψ^* *a fragmentation such that* $\hat{\mathcal{K}} \supset \{B_1, \ldots, B_n\}$, *each* B_i *being the behavior* $Bhv(\wp(\psi_i))$, $\wp(\psi_i) = (\psi_{i_0}, \mathcal{V}_i, \mathcal{O}_i, \mathcal{R}_i, \mathcal{K}_i)$, $\mathcal{V}_i \Subset \hat{\mathcal{V}}_{\langle\psi_i\rangle}$, $\mathcal{O}_i \ni (\hat{\mathcal{O}}_{\langle\psi_i\rangle})_{[\mathcal{V}_i]}$, *and* $\psi_0 = (\psi_{1_0}, \ldots, \psi_{n_0})$. *Let* N_0 *be the root of the diagnostic graph* $D^{\mathrm{g}} = Dgraph(Dtree(\hat{\wp}(\psi), \psi^*))$. *Then,*

$$\Delta\big(\hat{\wp}(\psi)\big) = \Delta\big(Abd\big(Bhv^{\mathrm{g}}(N_0), \hat{\mathcal{R}}\big)\big). \tag{80}$$

Example 30. With reference to the system Ξ displayed in Fig. 20 (p. 271), consider the diagnostic problem $\wp(\Xi) = (\Xi_0, \mathcal{V}, \mathcal{O}, \mathcal{R}, \mathcal{K})$ defined in Example 28. A relevant diagnostic tree D^{t} is displayed in Fig. 22 (p. 273), while relevant observations and viewers are outlined in Fig. 21 (p. 272) and Table 2 (p. 272), respectively. The assumptions made in Example 28 offer evidence that Proposition 17 is applicable to the solution of $\wp(\Xi)$, which requires four steps, namely (i) the generation of a diagnostic graph D^{g} relevant to D^{t}, (ii) the computation of the Bhv^{g} function on the root of D^{g}, (iii) the creation of the relevant abduction, and, finally, (iv) the extraction of the set of candidate diagnoses from the latter.

Since the diagnostic graph D^{g} has been generated already (see Fig. 23, p. 275), the essential problem is to compute the behavior relevant to $\wp(\Xi)$ by means of the function $Bhv^{\mathrm{g}}(N_0)$, where N_0 is the root of D^{g}. The subsequent steps (iii)–(iv) can be carried out as shown in Section 8.

The recursive nature of Bhv^{g} requires us to compute the behavior of Ξ bottom-up within D^{g} (Fig. 23). A possible computation involves the following sequence of nodes $\langle N_7, N_3, N_4, N_1, N_6, N_2, N_0\rangle$. Based on Eq. (74), $Bhv^{\mathrm{g}}(N_7)$ is simply the behavior associated with ξ_1 in the compiled knowledge.

By contrast, the computation of Bhv^{g} for the remaining nodes translates to the computation of the *Join* function (see Eq. (47)) applied to the set of (possibly projected) behaviors associated with the child nodes, based on the observation and viewer of the parent node. Specifically, the behavior relevant to N_3 will be

$$Bhv^{\mathrm{g}}(N_3) = Join\big(\big\{Bhv^{\mathrm{g}}(N_7)\big\}, \mathcal{V}_1, \mathcal{O}_1\big),$$

where the set of child behaviors to be joined is in fact a singleton. The resulting behavior is displayed on the top (left-hand side) of Fig. 24 (p. 278). Note how each node is identified by a pair $(n, \Im)$, where n is a node in Fig. 4 (p. 241) and $\Im$ is a node of the index space relevant to $\mathcal{O}_1$ in Fig. 21 (first column). Intuitively, $Bhv^{\mathrm{g}}(N_3)$ incorporates the subset of histories in Fig. 4 that comply with $\mathcal{V}_1$ and $\mathcal{O}_1$. A node is final when both n and $\Im$ are final in the respective graphs. Nodes of the resulting graph are in turn identified with numbers $(0 \ldots 11)$. In a similar way, the behavior relevant to N_4 will be

$$Bhv^{\mathrm{g}}(N_4) = Join\big(\big\{\big(Bhv^{\mathrm{g}}(N_7)\big)_{[\xi_2]}\big\}, \mathcal{V}_2, \mathcal{O}_2\big),$$

where $Bhv^{\mathrm{g}}(N_7)$ is projected on ξ_2 (see Eq. (65), p. 274). The resulting behavior is shown in Fig. 24, next to $Bhv^{\mathrm{g}}(N_3)$. The same applies to N_6, namely

$$Bhv^{\mathrm{g}}(N_6) = Join\big(\big\{Bhv^{\mathrm{g}}(N_7)_{[\xi_4]}\big\}, \mathcal{V}_4, \mathcal{O}_4\big),$$

whose resulting behavior is shown next to $Bhv^{\mathrm{g}}(N_4)$. The computation of the behavior associated with N_1, namely

$$Bhv^{\mathrm{g}}(N_1) = Join\big(\big\{Bhv^{\mathrm{g}}(N_3), Bhv^{\mathrm{g}}(N_4)\big\}, \mathcal{V}_5, \mathcal{O}_5\big),$$

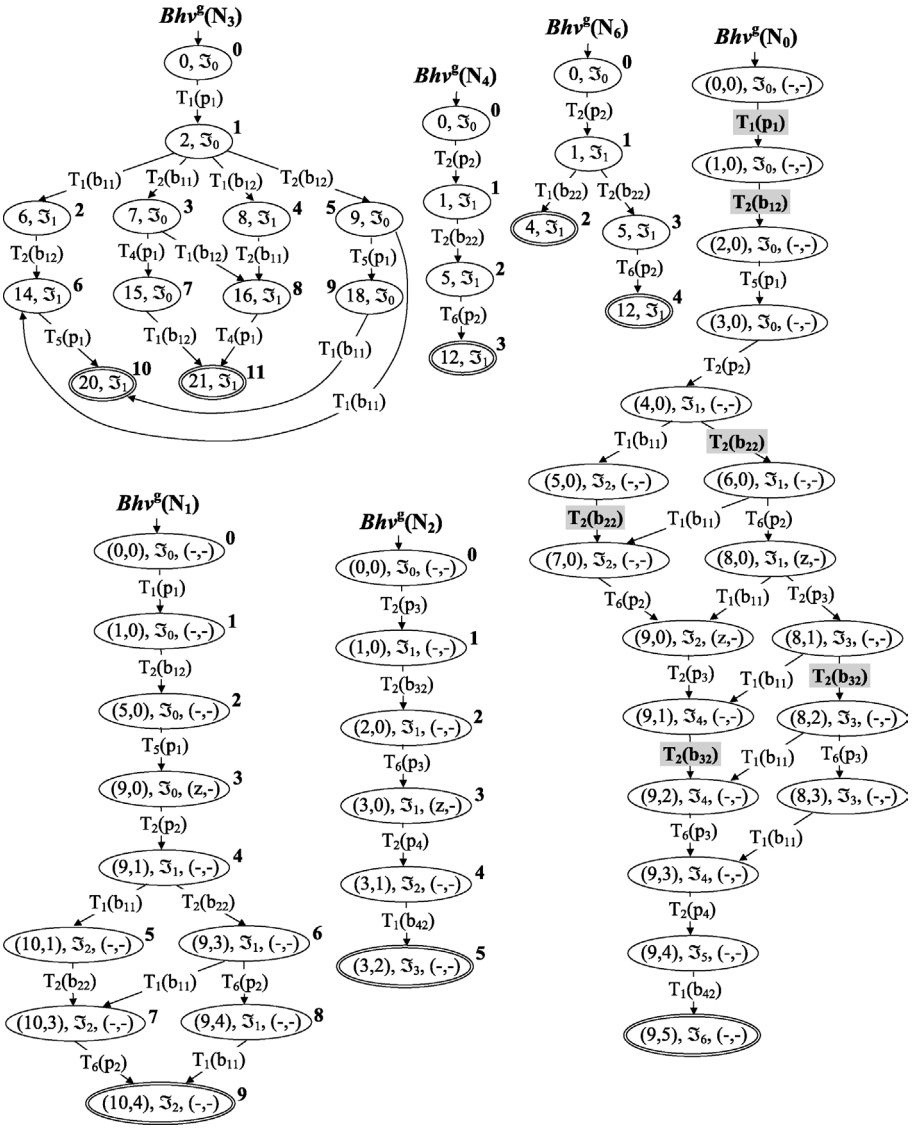

Fig. 24. Behaviors relevant to the internal nodes of the diagnostic graph in Fig. 23.

does not involve any projection, as the child nodes of N_1 are the same as in the diagnostic tree.[26] The resulting graph is shown on the bottom (left-hand side) of Fig. 24. Note how nodes are identified by a triple $(\sigma, \Im, Q)$, where $\sigma = (n, n')$ is a pair of nodes in $Bhv^{g}(N_3)$

[26] Behavior projection is required whenever the edge from the parent node to the child node is marked by a system that differs from the system associated with the child node.

and $Bhv^{g}(N_4)$, respectively, $\Im$ is a node of the index space $Isp(\mathcal{O}_5)$ (fifth column in Fig. 21), while Q is the pair of queues of events within the two links connecting ξ_1 with ξ_2 (see Fig. 20, p. 271). In this case, a node is final when n, n', $\Im$ are final and queues in Q are empty. The behavior associated with N_2 is determined by

$$Bhv^{g}(N_2) = Join\big(\big\{\big(Bhv^{g}(N_4)\big)_{[\xi_3]}, Bhv^{g}(N_6)\big\}, \mathcal{V}_6, \mathcal{O}_6\big),$$

where the child behavior of N_4 is projected on ξ_3. The resulting graph is depicted in the center (bottom) of Fig. 24. As for N_1, each node $(\sigma, \Im, Q)$ is such that $\sigma = (n, n')$, where n and n' are nodes of $Bhv^{g}(N_4)$ and $Bhv^{g}(N_6)$, respectively, $\Im$ is a node of the index space $Isp(\mathcal{O}_6)$ (sixth column in Fig. 21), and Q is the pair of queues of events within the two links connecting ξ_3 and ξ_4 (Fig. 20).

Eventually, the behavior associated with the root is made up by joining the behaviors relevant to N_1 and N_2 based on observation $\mathcal{O}$ and viewer $\mathcal{V}$:

$$Bhv^{g}(N_0) = Join\big(\big\{Bhv^{g}(N_1), Bhv^{g}(N_2)\big\}, \mathcal{V}, \mathcal{O}\big).$$

In the resulting graph displayed on the right-hand side of Fig. 24, each node $(\sigma, \Im, Q)$ is such that $\sigma = (n, n')$, where n and n' are nodes of $Bhv^{g}(N_1)$ and $Bhv^{g}(N_2)$, respectively, $\Im$ is a node of the index space $Isp(\mathcal{O})$ (right-hand side of Fig. 21), and Q is the pair of queues of events within the two links connecting ξ_5 with ξ_6.

Within the pictorial representation of $Bhv^{g}(N_0)$, faulty transitions are shaded. This allows us to have an idea on how the corresponding abduction looks like. In particular, it is easy to decorate each node of the behavior with the relevant set of diagnoses and find out that the solution of the diagnostic problem is the singleton

$$\Delta\big(\wp(\Xi)\big) = \Delta\big(Abd\big(Bhv^{g}(N_0)\big)\big) = \big\{\{s_1, f_{12}, f_{22}, f_{32}\}\big\}.$$

Based on the system Ξ depicted in Fig. 20 and the ruler $\mathcal{R}$ defined in Example 28 (p. 271), this result can be phrased as follows: *A short circuit occurred to device D_1 and breakers b_{12}, b_{22}, and b_{32} failed to open.*

In this paper we do not face the problem of generating a diagnostic tree, which should be based on optimization criteria.[27] Instead, we give an algorithm for yielding a diagnostic graph of a given diagnostic tree, as detailed below.

Algorithm 3. The *Abridge* function generates a diagnostic graph D^{g} relevant to a given diagnostic tree D^{t}. Basically, it searches for groups of isomorphic nodes and substitutes each group with one of the nodes. This replacement requires a rearrangement of the edges too. Specifically, the set of nodes of D^{g} is initialized as a copy of the set of nodes of D^{t} (line 1). Instead, the set of edges of D^{g} is obtained by marking each edge of D^{t} with the subsystem relevant to the node entered by the edge (line 2). The core of the algorithm is

[27] The problem of generating a diagnostic tree is similar to the problem of generating a query plan in a relational database system, where the given SQL query is mapped onto an expression of Relational Algebra represented by a tree [22]. Specifically, a diagnostic tree corresponds to a query plan. In spite of the different formal domains (diagnosis vs. databases), the basic problem is the same, namely yielding an 'optimized tree' in order to speed up the relevant computation (query execution vs. behavior generation).

represented by the loop within lines 4–12. At each iteration, an unprocessed node N is picked up from $\mathbf{N}^{\mathrm{g}}$ (line 5), and the set $\mathbb{N}$ of nodes isomorphic to N is determined (line 6). A node N^{g} is chosen among such isomorphic nodes (line 7) and all edges entering all other nodes in $\mathbb{N}$ are redirected towards N^{g} and remarked (lines 8–10). Before ending the iteration, all nodes in $\mathbb{N}$ are marked (line 11). When all nodes have been processed (exiting condition at line 12), all nodes that are not connected with the root are removed from the graph (line 13), as well as the resulting dangling edges (line 14).

function *Abridge*(D^{t})
  input
    $D^{\mathrm{t}} = Dtree(\hat{p}(\Psi), \Psi^*) = (\mathbf{N}^{\mathrm{t}}, \mathbf{E}^{\mathrm{t}}, N_0)$: a diagnostic tree;
  output
    $D^{\mathrm{g}} = Dgraph(D^{\mathrm{t}}) = (\mathbf{N}^{\mathrm{g}}, \mathbf{E}^{\mathrm{g}}, N_0)$: a diagnostic graph relevant to D^{t};
  begin
1. $\mathbf{N}^{\mathrm{g}} := \mathbf{N}^{\mathrm{t}}$;
2. $\mathbf{E}^{\mathrm{g}} := \{N \xrightarrow{\psi'} N' \mid N \to N' \in \mathbf{E}^{\mathrm{t}}, N' = (\psi', \psi'_0, \mathcal{V}', \mathcal{O}')\}$;
3. Mark node N_0 in $\mathbf{N}^{\mathrm{g}}$;
4. **repeat**
5. Get an unmarked node $N \in \mathbf{N}^{\mathrm{g}}$;
6. $\mathbb{N} := \{N' \mid N' \in \mathbf{N}^{\mathrm{g}}, N' \doteq N\}$;
7. Choose a node $N^{\mathrm{g}} = (\psi^{\mathrm{g}}, \psi_0^{\mathrm{g}}, \mathcal{V}^{\mathrm{g}}, \mathcal{O}^{\mathrm{g}}) \in \mathbb{N}$;
8. **for each** edge $N_1 \xrightarrow{\psi_2} N_2 \in \mathbf{E}^{\mathrm{g}}$ such that $N_2 \in (\mathbb{N} - \{N^{\mathrm{g}}\})$ **do**
9. $\mathbf{E}^{\mathrm{g}} := (\mathbf{E}^{\mathrm{g}} - \{N_1 \xrightarrow{\psi_2} N_2\}) \cup \{N_1 \xrightarrow{\psi_2} N^{\mathrm{g}}\}$
10. **end-for**;
11. Mark each node in $\mathbb{N}$
12. **until** all nodes in $\mathbf{N}^{\mathrm{g}}$ are marked;
13. Remove from $\mathbf{N}^{\mathrm{g}}$ all nodes that are not reachable from N_0;
14. Remove from $\mathbf{E}^{\mathrm{g}}$ all the dangling edges
  end.

Example 31. The diagnostic graph D^{g} displayed in Fig. 23 (p. 275) can be generated from the diagnostic tree D^{t} shown in Fig. 22 (p. 273) by means of the *Abridge* function as described in Table 3. Each row of the table outlines the significant information associated with each iteration of the loop enclosed in lines 4–12. Specifically, *Marked* is the set of marked nodes in $\mathbf{N}^{\mathrm{g}}$ at the beginning of the iteration. N is the unmarked node that is picked up from $\mathbf{N}^{\mathrm{g}}$ (line 5). $\mathbb{N}$ is the set of nodes in $\mathbf{N}^{\mathrm{g}}$ that are isomorphic to N (line 6). N^{g} is the node picked up from $\mathbb{N}$ (line 7). With reference to line 9, $-\mathbf{E}^{\mathrm{g}}$ is the set of edges removed from $\mathbf{E}^{\mathrm{g}}$, while $+\mathbf{E}^{\mathrm{g}}$ is the set of edges replacing the former.

For example, at the beginning of the fourth iteration, the topology of D^{g} is isomorphic to D^{t}, as no edge has been removed. The set of marked nodes includes $N_0 \ldots N_3$ and the unmarked node N chosen from $\mathbf{N}^{\mathrm{g}}$ is N_5. The set $\mathbb{N}$ of nodes isomorphic to N_5 involves $N_4 \ldots N_6$, among which $N^{\mathrm{g}} = N_4$ is chosen. Since $\mathbb{N}$ is not a singleton, the loop within lines 8–10 is iterated once, namely for the edge from N_2 to N_5, thereby substituting such edge, namely $-\mathbf{E}^{\mathrm{g}}$, with $+\mathbf{E}^{\mathrm{g}}$, that is, $N_2 \xrightarrow{\xi_3} N_4$. This results in the dangling of node N_5,

Table 3
Computation of the diagnostic graph of Fig. 23 by means of Algorithm 3

Iteration	Marked	N	$\mathbb{N}$	N^{g}	$-\mathbf{E}^{\mathrm{g}}$	$+\mathbf{E}^{\mathrm{g}}$
1	$\{N_0\}$	N_1	$\{N_1\}$	N_1	$\emptyset$	$\emptyset$
2	$\{N_0, N_1\}$	N_2	$\{N_2\}$	N_2	$\emptyset$	$\emptyset$
3	$\{N_0, N_1, N_2\}$	N_3	$\{N_3\}$	N_3	$\emptyset$	$\emptyset$
4	$\{N_0, N_1, N_2, N_3\}$	N_5	$\{N_4, N_5\}$	N_4	$\{N_2 \xrightarrow{\xi_3} N_5\}$	$\{N_2 \xrightarrow{\xi_3} N_4\}$
5	$\{N_0, N_1, N_2, N_3, N_4, N_5\}$	N_6	$\{N_6\}$	N_6	$\emptyset$	$\emptyset$
6	$\{N_0, N_1, N_2, N_3, N_4, N_5, N_6\}$	N_{10}	$\{N_7, N_8, N_9, N_{10}\}$	N_7	$\{N_4 \xrightarrow{\xi_2} N_8,$ $N_5 \xrightarrow{\xi_3} N_9,$ $N_6 \xrightarrow{\xi_4} N_{10}\}$	$\{N_4 \xrightarrow{\xi_2} N_7,$ $N_5 \xrightarrow{\xi_3} N_7,$ $N_6 \xrightarrow{\xi_4} N_7\}$

which is no longer connected with N_2 (the relevant edge has been redirected toward N_4 and marked anew).

A similar processing is carried out at the sixth iteration, where node N_7 is chosen among the leaf nodes $N_7 \ldots N_{10}$. In this case, the redirected edges are those leaving N_4, N_5, and N_6, thereby causing nodes N_8, N_9, and N_{10}, respectively, to be dangling.

Therefore, at the end of the loop, the nodes in $\mathbf{N}^{\mathrm{g}}$ that are not reachable from the root are N_5, N_8, N_9, and N_{10}, which are therefore removed (line 13). The algorithm terminates by removing the dangling edge $N_5 \xrightarrow{\xi_3} N_7$.[28]

A final note is worthwhile on fragmented problems. One may ask why the fragmented approach (grounded on Requirement 5) is confined to behaviors only, while abductions and maps are not considered. So, why not joining abductions or maps to make up the global knowledge graph starting from such fragmented knowledge? More generally, why not considering a *hybrid* fragmented problem where different sorts of knowledge are available for different subsystems rather than the same sort for all subsystems? Here are some comments relevant to these questions.

A first point, which refers to maps, is that maps cannot be merged when we assume system ψ to be connected, that is, when the fragmentation ψ^* involves subsystems that are connected to one another through links.[29] The reason lies in the impossibility for maps to combine histories from subsystems, because maps only refer to observable labels and not

[28] Algorithm 3 may be improved by discarding any processing over nodes and edges that are no longer connected with the root owing to a redirection at line 9. For example, at the fourth iteration, the redirection of an edge leaving node N_2 causes N_5 to be dangling. Hence, all the part descending such node (including N_9 and the entering edge) might be eliminated from D^{g} at once.

[29] When ψ^* is composed of disconnected subsystems, the solution of the problem relevant to ψ can be trivially generated by combining the solutions of the subsystems. For example, assuming the same ruler, if $\psi^* = \{\psi_1, \psi_2\}$, then $\Delta(\wp(\psi)) = \{\delta \mid \delta = \delta_1 \cup \delta_2, \delta_1 \in \Delta(\wp(\psi_1)), \delta_2 \in \Delta(\wp(\psi_2))\}$.

to component transitions, the latter being essential for computing the candidate diagnoses of $\wp(\psi)$.

A second point refers to abductions. We can join a set of abductions relevant to a fragmentation ψ^* as follows. Consider the simple case $\psi^* = \{\psi_1, \psi_2\}$, with corresponding abductions A_1 and A_2. The *Join* operator defined in Section 11 might be naturally extended to generate the abduction A relevant to ψ by considering states $(\mathbb{A}, \Im, \mathbb{L}, \delta)$ of A, where $\mathbb{A}$ is the pair of states of A_1 and A_2, $\Im$ and $\mathbb{L}$ maintain the same meaning, while δ is the associated diagnosis. For example, assume $\mathbb{A} = (S_1, S_2)$, where $S_1 = (\beta_1, \delta_1)$ and $S_2 = (\beta_2, \delta_2)$, and a transition $S_1 \xrightarrow{T} S_1'$ in A_1, where $S_1' = (\beta_1', \delta_1')$. This determines a transition $(\mathbb{A}, \Im, \mathbb{L}, \delta) \xrightarrow{T} (\mathbb{A}', \Im', \mathbb{L}', \delta')$ in A, where $\mathbb{A}' = (S_1', S_2)$, $\Im'$ and $\mathbb{L}'$ are computed as usually, while $\delta' = \delta_1' \cup \delta_2$.[30] However, considering the consecutive join operations relevant to a diagnostic graph and the fact that an abduction is, in the general case, larger than a behavior (due to possible replications of the same behavior state in different states of the abduction), we cannot take it for granted that joining abductions is better than joining behaviors upward in the diagnostic graph (by confining the generation of the abduction to the root of the graph).

A final point concerns hybrid fragmented problems, where both behaviors and abductions are available. In this case, at each joining step, we can first perform a sort of coercion of behaviors into abductions and then join all abductions.

To summarize, fragmented diagnostic problems cannot be extended to maps when ψ is connected. However, they can be extended to abductions in a natural way. More generally, we can consider hybrid fragmented problems, where both behaviors and abductions are involved in the same fragmentation.

12. Discussion

This paper copes with *a-posteriori* diagnosis of a class of asynchronous DESs. The a-posteriori attribute means that the input observation encompasses a whole history, this being a sequence of component transitions that moves the system from an initial state to a final state, where in both such states all links are empty.

The either transient or persistent nature of detected faults does not depend on the processing method, rather on the way each fault is assigned (by the ruler) to the transitions in the relevant component models. For instance, a fault assigned to transition T_2 of the breaker model in Fig. 2 (p. 238) is transient since it leads to a state where a completely normal behavior is still possible. A persistent fault should typically be assigned to a transition leading to a state where the operating modes of the relevant component are irreversibly degraded or diminished. Several modeling styles, both for components and rulers, however, can be adopted to model the same behavior, where a style is not just a matter of taste but of convenience and intuitiveness also. The diagnostic method, far from imposing a univocal modeling style, is required to automatically find out the occurred faults, given the component models and the ruler. If a component fault is included in a (deep) diagnosis, this

[30] δ' can also be computed as $\delta \cup \{\varphi\}$, where φ is the faulty label of T based on the ruler of $\wp(\psi)$.

means that such a fault occurs at least once during a history of the system consistent with the observation. Any further interpretation is not up to the diagnostic method, instead it can be carried out only by the modeler. The semantics of a candidate diagnosis is that each fault belonging to it has occurred to the system (but may not affect the system at present).

A remark is worthwhile about the proposed fragmented approach to a-posteriori diagnosis. Based on Corollary 14.2, there may be several diagnostic trees decomposing the same problem, all of which lead to the same solution but with different costs, as anticipated in [23] and proven in [24], where a limited notion of a diagnostic tree is adopted. Enforcing optimization criteria while building the diagnostic tree for the problem at hand is still an open issue and a wide topic for future research. In the meantime, [24] shows how to exploit the same hierarchical decomposition adopted in the compositional topological model of the system also for solving fragmented problems.

The present work differs from the other ones in the domain of model-based diagnosis of DESs in several major respects. A novelty consists in decoupling the (behavioral) models of system components from the descriptions of their observability and abnormality properties. In the literature, only a more limited separation from observability properties can be found: in [6] each specific problem assigns the same observability to all the instances of the same component type, whereas in our approach each instance can be endowed with distinct properties. It is worth noting that [6] gives the conceptual means to *characterize* model-based diagnosis, not to compute diagnoses, whereas our proposal is aimed at providing also operational methods. As to the separation from abnormality properties, this is an exclusive feature of the approach described in the present paper.

In the literature, knowledge compilation has so far been confined to an off-line activity only, whose result, compiled knowledge, can be used for solving any diagnostic problem relevant to the system such knowledge refers to. In flexible similarity-based diagnosis, knowledge can be compiled also on-line and the suitability of a chunk of knowledge to solve a given problem has to be checked based not only on the system, but on the whole considered problem. In addition, previous contributions face isomorphism only intuitively, without never mentioning it explicitely, and just at the topological level [14,25,26], without showing how to detect isomorphic structures. The current work, instead, both extends the concept of isomorphism to diagnostic problems and provides a formal method to detect isomorphic problems, i.e., to recognize in advance, given the specifications of two problems, whether the solution of the former can be mapped to that of the latter by means of a renaming operation. This ability translates to an increase in efficiency of both knowledge compilation and exploitation. Moreover, no approach by other authors adopts a concept of an observation as expressive as that of the current paper.

In the following, some approaches to diagnosis of DESs in the literature are briefly compared with the current work for isolating further differences besides the ones highlighted above, which all hold for each of them.

12.1. Diagnoser approach

In the diagnoser approach [15,16] off-line activities process the behavioral models of system components in order to produce a global *system model* and, then, draw a *diagnoser* from it. The diagnoser is a deterministic, completely observable FSM, against which the

system observation is checked on-line to infer about past occurrences of failures. Diagnosing a DES amounts to following on-line a path in the diagnoser, starting from its (normal) initial state, and taking for each observed event the (only) edge marked by the event itself, thus reaching a node that contains all the possible current states of the DES, where each state is associated with a set of faults.

If the same state in the system model can be reached via two or more distinct paths, producing the same observation and characterized by distinct set of faults, then the set of faults associated with the state is the intersection of all such sets, to which a label is added for denoting ambiguity, which, indeed, means incompleteness of candidate diagnoses. While (off-line) constructing the diagnoser, this ambiguity is propagated to the successor states of the ambiguous state in the next nodes as far as no faulty transition is involved. Once a faulty transition has been encountered, the set of faults is updated and the ambiguity label is dropped (although the relevant candidate diagnoses keep on being incomplete).

As underlined in [5], building the global system model, although via a well known operation of synchronization (which, indeed, is not enough in general since also the sensor map has to be considered), is unrealistic due to its intractable size for large systems. Moreover, in the worst case, the state space of the diagnoser is exponential in the state space of the system model [15]. Hence, also the diagnoser cannot be produced. By contrast, the active system approach has always avoided generating any global system model as well as any global diagnoser. The map space of the current approach may resemble the diagnoser of the diagnoser approach, however the following statements highlight some meaningful differences.

(1) A map space is meant to a-posteriori diagnosis, while the diagnoser is meant to diagnosis during monitoring: this translates to codifying a different set of diagnoses within each node, that is, the candidate diagnoses inherent to all the possible current states in the diagnoser, and the candidate diagnoses inherent to final states only in the map space;
(2) As a consequence of the previous point, a node in a map space consists of a set S of (abduction space) states and in a set $\mathbb{D}$ that gathers all the diagnoses inherent to the final states in S; in a diagnoser, instead, each node consists in a set of pairs, each pair being a system state and its relevant diagnosis. In other words, in a map space, unlike a diagnoser, the association state-diagnosis is not recorded, thus reducing considerably the space complexity of the map space with respect to the diagnoser;
(3) A map space can be generated by means of a modular strategy that reduces the size of the search space and is amenable to a parallel execution, while no modular method for the generation of the diagnoser has been proposed;
(4) A map space can be used for a class of isomorphic systems (as well as for any of their subsystems, even if this aspect has not been stressed in the paper), while a diagnoser is relevant to a single system;
(5) The set of diagnoses computed by processing a map space is complete while completeness of diagnoses does not hold in general in the diagnoser approach;
(6) A system can be diagnosed without generating or processing any map space in the flexible diagnosis approach, while a diagnoser is strictly necessary for solving any diagnostic problem in the diagnoser approach.

The diagnoser and the active system approach traditionally consider two distinct subsets of DESs: synchronous and asynchronous, respectively (as is also in this paper, although in other contributions the latter has extended its interest to *polymorphic* DESs, that integrate both synchronous and asynchronous behavior [19,27]). Moreover, the diagnoser approach adopts a completely certain observation.

12.2. Extended diagnoser approach

In [26], a work rooted in [14], the diagnoser approach [15,16] is adapted to deal with telecommunication networks. Analogously to the diagnoser approach, a behavioral model of the system is produced off-line and transformed into an *extended diagnoser*, an automaton against which observed events are matched on-line. The adopted compositional model of a system is quite similar to that of the active system approach and different from the diagnoser approach since connections between components are modeled as asynchronous.

A limitation addressed by [26] is the incompleteness of the output of the diagnoser approach when a system is taken into account that is not diagnosable (according to the definitions of diagnosability and I-diagnosability provided by the diagnoser approach). The price paid in [26] for achieving the completeness of candidate diagnoses is the nondeterminism of the extended diagnoser vs. the determinism of the original diagnoser, that is, there may be several distinct edges leaving a state of the extended diagnoser marked with the same observable event. This means that a nondeterministic search is performed on-line when matching the observation against the extended diagnoser. The completeness of candidate diagnoses is guaranteed also by the approach in this paper (and by the active system approach in general), while the search in a map space is more efficient, since deterministic.

Moreover, the extended diagnoser approach deals with explicit temporal information, which is completely missing in the original approach. In particular, it integrates temporal constraints concerning emission and transmission delays on connections, which are important for relaxing the hypothesis of permanent failures made by the diagnoser approach, as in the considered case study there are only transient failures (and comebacks take some time). To this end, internal events have been introduced, which are unobservable events used to model non instantaneous changes of states.

To cope with the computational difficulties of the diagnoser approach, deriving from the need for generating the whole system model, in [26] a *generic* model of a subsystem is built, from which a *generic* (extended) diagnoser is drawn. Doing so means taking advantage of the regular structure of the system in both the modeling and the diagnostic task. The latter, in fact, provides diagnoses inherent to all the distinct instances of a modeled subsystem (called generic component) and, in addition, it produces the diagnostic output in a concise way by exploiting set-theoretic operators. All these advantages, however, can be obtained when a system consists of several topologically identical subsystems not interacting with each other, as the telecommunication network at hand, but does not solve the problem of scalability in the general case, wherein the interaction between subsystems has to be managed as well as the consistency of candidate diagnoses in case there are components shared by two or more subsystems. Moreover, even in case the system can be viewed as an assembly of topologically identical non-interacting subsystems, the issue of scalability shifts from the system to the subsystem, which may itself be too large for the

generation of its generic model, or, still more, for drawing the generic diagnoser from the generic model. In fact, given the same initial system model, the corresponding extended diagnoser is larger than the diagnoser of the original diagnoser approach.

Summing up the main differences between the approach in [26] and the one described in this paper, first of all the former performs diagnosis while monitoring the system whereas the task performed by the latter is a-posteriori diagnosis. Moreover, the observation taken as input by the extended diagnoser approach is a completely certain stream of time-stamped alarms and represents time constraints explicitly, while our approach takes into account temporal precedence relationships.

In order to diagnose a (sub)system, the generation of its behavioral model is needed in [26] while it is not necessary in our approach. The algorithm for building such a model is not modular in [26] while all algorithms proposed in the active system approach for behavior reconstruction are modular.

In the extended diagnoser approach all faults are transient. Instead, in the active system approach each fault may be either permanent or transient and the management is uniform for all of them. In addition, (topological) isomorphism in [26] is considered as a means of reducing the reasoning effort, while (a broader notion of) isomorphism is a means of achieving reuse in the current approach.

12.3. Chronicle approach

A whole family of knowledge-compilation approaches to diagnosis of DESs is automatic chronicle generation based on simulation [25,28]. A *chronicle* is the temporal pattern of observable events that the considered system is expected to output along one or more evolutions corresponding to a given (normal/abnormal) situation. A chronicle recognition tool, which is provided with a set of chronicles, is in charge of analyzing the stream of observable events produced by the system in order to identify, on the fly, the occurrence of any instance of a chronicle. Chronicle recognition by (on-line) processing each observed event is linear with the number of chronicles, therefore, chronicle-based approaches are more efficient than model-based approaches (performing abductive/consistency-based reasoning on-line).

The technique for automatic chronicle generation consists in the off-line simulation of the behavioral model of the system, aimed at collecting a set of sequences of (possibly time-stamped) observable events resulting from well-identified situations. This set is taken as input by a learning module that generates the set of discriminating chronicles to be exploited on-line. The simulation required by the chronicle-learning process corresponds to the behavior reconstruction in the active system approach. The model of the system to be diagnosed described in [25], is a set of nondeterministic automata, which, unlike the active system approach, do not communicate through buffers. Another difference with respect to our work is that in [25] a time interval can be associated with each transition, as in [14,26], representing an (uncertain) delay in emitting the output message. This raises the problem of exhaustively simulating such a temporal model, which is still open in the general case [29]. Simulation consists in triggering (single or multiple) fault events in the system model and propagating the generated events. Due to uncertainty on delays, one fault can yield several sequences of observed events, differing in the order of the events and

even in the content of such sequences when a masking phenomenon occurs. The simulation performed by our approach for generating compiled knowledge is atemporal, however the buffered communication supported by the active-system topology allows for reconstructing histories that produce sequences of observed events corresponding to all possible relative delays. Moreover, behavioral models endowed with uncertain events, introduced in [20] but not dealt with in this paper, allow for the representation of masking phenomena.

12.4. Decentralized diagnoser approach

An attempt to combine the use of a diagnoser (generated off-line), as in the diagnoser approach [15,16], with the ability to adopt (on-line only) a divide-and-conquer strategy, as in the original active system approach [10], is performed in [3]. Similarly to the active system approach, no global behavioral model is needed, instead a local diagnoser is drawn off-line for each component, this being an automaton whose states and (observable) transitions are labeled with compiled knowledge about unobservable paths and interacting components, respectively. Each local diagnoser is employed on-line for both reconstructing the possible evolutions of the relevant component that comply with the observation and merging the 'local' histories of distinct components into global system histories. However, being nondeterministic, a local diagnoser does not bring a substantial increase in on-line efficiency while building local histories with respect to the use of a component model, which is the case in the original active system approach as well as in (on-line) model-based problem solving and (off-line) compilation of general purpose knowledge in the current approach.

The merging algorithm adopted by [3] is basically the same as in the active system approach [10], the only difference being that the reconstruction plan is not generated beforehand, as is still in the current work, instead it is built on the fly after the reconstruction of the local histories has been completed, so as to exploit the information about component interactions implicit in the local histories. This choice, aimed at reducing the size of the search space, has both pros and cons, as discussed in [20]. By contrast, in flexible diagnosis, the efficiency of both off-line and on-line processing benefits from a recursive modular technique and compiled knowledge exploited on-line does not necessarily refer to a single component, rather it may be inherent to a whole cluster of components.

The approach described in [3] tackles synchronous systems only, the same as the diagnoser approach, while the method presented in this paper addresses asynchronous systems. Finally, the diagnoses provided by the approach in [3] are histories while those of the current diagnostic method are deep diagnoses.

13. Conclusion

This paper proposes both the modeling primitives and the reasoning mechanisms for performing a-posteriori diagnosis of a class of DESs in a flexible way. A novelty consists in confining the observability and abnormality properties of the system components within two problem-level modeling entities, the viewer and the ruler. This separation of concerns is worthwhile in several aspects. The specification of a ruler allows for the customization of the diagnostic output, depending on the specific diagnostic session, where different de-

grees of precision may be required in different diagnostic problems for the same system. The advantage of separation of concerns in observability is twofold. On the one hand, viewers allow different diagnostic tasks to run on the same system with different observability conditions, namely they support dynamic observability, which is essential when the sensor/communication apparatus is time-varying. On the other, the notion of a viewer is bound to support the open problem of diagnostic-tree optimization, as the subproblems relevant to the internal nodes of the tree may be conveniently formulated by choosing viewers that maximize the chances of factorization for the equivalent diagnostic graph. Specifically, it might be convenient to 'adapt' a viewer in a certain node (preserving the subsumption conditions with child nodes) so as to obtain an isomorphism with the problem relevant to another node, thereby 'forcing' node factorization. Analogous considerations apply to the ruler. Diagnostic-tree optimization can be achieved by manipulating the nodes of the diagnostic tree, however no such manipulation is described in the present paper.

Specifically, the function proposed in this paper replaces each set of isomorphic diagnostic-tree nodes with a single node, that is, the same subproblem in the hierarchical decomposition is solved only once. Thus the node factorization degree affects the computational savings. Node factorization depends on the 'regularity' of the problem features, thus 'regular' problems are those solved more efficiently by flexible diagnosis. For instance, the more regular the system topology, i.e., the more defined in terms of isomorphic subsystems, the more likely the factorization, the more efficient the solution of the fragmented problem. However, based on past research on modular diagnosis of active systems [10,18], evidence shows that, although regularity is important for the factorization of the diagnostic tree, the hierarchical solution of a fragmented problem is in general more efficient than a single join operation of the behaviors relevant to the fragmentation, even without any factorization. When no factorization is possible, the advantage comes from applying the constraints imposed by the topology and the observation to subsystems where components are tightly coupled.

The proposed modular method can be used both off-line and on-line. Modularity is, in principle, synergetic with reusability since the hierarchical decomposition of the knowledge generation process can be performed in such a way that the knowledge required by one or more nodes is already available from previous processing.

Reusability applies to every piece of knowledge exploitable for diagnostic-problem solving. Two epistemological kinds of knowledge are distinguished, general-purpose and special-purpose, the latter being typically computed on-line since, unlike the former, depends on the observation. Knowledge of either kind is further classified based on its independence from viewers and/or rulers, thus featuring several distinct levels of generality. The paper provides the means to both generate and exploit each of the above types of knowledge for solving specific problems. Any piece of knowledge that is independent of any ruler can be obtained by processing the behavioral models of system components. General knowledge about a system can be used for solving any problem involving such a system or any isomorphic one. Specific knowledge, instead, is exploitable for solving a given problem only under given subsumption conditions. This points out that a chunk of knowledge meant to solve a specific problem can support the solution of 'analogous problems' too. As such, the approach is the basis for carrying out the task of diagnosis by means of a kind of analogical reasoning mechanism.

Solving a diagnostic problem amounts to processing existing knowledge relevant to the given system provided such knowledge contains either directly, if the knowledge is specific, or indirectly, if not, all the evolutions belonging to the solution. The more specific the compiled knowledge, the less computationally expensive its on-line exploitation for solving a problem. If no compiled knowledge that subsumes the solution is available (or no compiled knowledge at all), the method resorts to the behavioral models of the system components, from which the solution of any problem can be drawn. Processing a piece of knowledge may lead to the generation of several increasingly specific pieces of knowledge, each of which can potentially be exploited in subsequent diagnostic sessions.

The proposed method includes, as a particular case, a previous approach by the authors [10], which relies on the availability of component models only. The diagnostic outputs of the approach are specific faults assigned to components rather than histories since drawing deep diagnoses from histories may be critical for humans, especially under strict time constraints.

Extending the current method so as to deal also with uncertain behavioral models as in [20] does not need further work since what changes is only the behavior reconstruction algorithm, which, however, is already available from past research. Updating the behavior reconstruction algorithm suffices also for extending the class of considered systems from synchronous to polymorphic [19]. Furthermore, the definition of simple restriction operations would allow a chunk of knowledge to be exploited for any subpart of the systems for which it was produced.

A challenge for future research is to provide an algorithm to choose the 'optimal' hierarchical fragmentation of the diagnostic problem. This should include a mixed strategy performing not only top-down but also bottom-up decomposition steps, so as to support knowledge reuse. Finally, techniques for continuous indexing and maintenance of knowledge are to be investigated.

Acknowledgement

We owe a debt of gratitude to our students, Simone Cerutti, Michele Scaroni, and Davide Zanni for their concrete support. We are most grateful to the anonymous reviewers, whose valuable comments and suggestions helped us improve the quality of the final manuscript.

Appendix A. Proofs

Proof of Proposition 1. (*Sketch*) Let $\tilde{\mathbf{B}}$ denote the subset of $\mathbf{B}$ including all and only the pairs having the transitions isomorphic to the pairs in $\mathbf{B}'$. As such, $\tilde{\mathbf{B}}$ and $\mathbf{B}'$ have the same cardinality, namely $n_{\mathbf{B}'}$. Consider the partition $Prt(\mathbf{B}')$ where each part contains all and only the elements of $\mathbf{B}'$ sharing the same label. Let $Prt(\tilde{\mathbf{B}})$ be the partition of $\tilde{\mathbf{B}}$ based on the same criterion. Checking whether there exists a binding subsumption $\mathbf{B} \ni \mathbf{B}'$ boils down to checking whether all the transitions in (any element within) a part $\tilde{b}_i$ of $Prt(\tilde{\mathbf{B}})$ are included in the same part of $Prt(\mathbf{B}')$. Finding out which is the part b'_i of $Prt(\mathbf{B}')$ that includes a given transition T requires at most a number of iterations that equals the number of pairs in $\mathbf{B}'$,

that is, in the worst case, $n_{\mathbf{B}'}$. Then the check of the remaining $(\tilde{n}_i - 1)$ transitions of part $\tilde{b}_i$ of $Prt(\tilde{\mathbf{B}})$ requires at most n'_i iterations for each transition, where $\tilde{n}_i$ and n'_i are the cardinalities of $\tilde{b}_i$ and b'_i, respectively. Thus, the cost of the check inherent to part $\tilde{b}_i$ is $n_{\mathbf{B}'} + (\tilde{n}_i - 1) \cdot n'_i$. Extending the check to all parts $\tilde{b}_i$ in $Prt(\tilde{\mathbf{B}})$ gives the complexity $\sum_i [n_{\mathbf{B}'} + (\tilde{n}_i - 1) \cdot n'_i]$, which equals $n^2_{\mathbf{B}'}$ when all parts $\tilde{b}_i$ are singletons and is a quadratic function of $n_{\mathbf{B}'}$ in the general case (which includes also the cost for generating partitions $Prt(\tilde{\mathbf{B}})$ and $Prt(\mathbf{B}')$). However, it is easy to find an implementation of the algorithm that is globally linear in $n_{\mathbf{B}'}$. □

Proof of Proposition 2. (*Sketch*) Eq. (15) is grounded on the intrinsic equivalence of $Isp'(\mathcal{O})$ and $Isp^{\mathrm{n}}(\mathcal{O})$. When the renaming introduces neither ε nor duplicates within the nodes of $\mathcal{O}$, Eq. (15) is supported by the fact that $Isp^{\mathrm{n}}(\mathcal{O})$ still represents all possible temporal sequences of $\mathcal{O}_{[\mathcal{V}']}$. If a label is mapped onto an ε, the same mapping will hold in $Isp^{\mathrm{n}}(\mathcal{O})$, with null effect on the temporal sequence. Finally, if two labels of the same node are mapped to a single label, the replication of the new label on two different edges of $Isp^{\mathrm{n}}(\mathcal{O})$ will result in a single edge after the transformation of $Isp^{\mathrm{n}}(\mathcal{O})$ into $Isp'(\mathcal{O})$. □

Proof of Proposition 3. (*Sketch*) Support for Proposition 3 is given in Example 14 (p. 253).

Proof of Proposition 4. (*Sketch*) Based on the definition, the solution of $\wp(\Psi)$ can be reformulated as $\Delta(\wp(\Psi)) = \{\delta \mid \delta = h \otimes \mathcal{R}, h \in Lang(Bhv(\wp(\Psi)))\}$. Each abduced diagnosis relevant to a history $h \in Lang(Abd(\wp(\Psi)))$ can be expressed as $\delta = \{\varphi \mid (T, \varphi) \in \mathcal{R}, T \in h\} = h \otimes \mathcal{R}$. Since $Lang(Abd(\wp(\Psi))) = Lang(Bhv(\wp(\Psi)))$, it follows that $\Delta(Abd(\wp(\Psi))) = \{\delta \mid \delta = h \otimes \mathcal{R}, h \in Lang(Bhv(\wp(\Psi)))\} = \Delta(\wp(\Psi))$. □

Proof of Proposition 5. (*Sketch*) The method for generating a behavior given an uncertain diagnostic problem, which can be either monolithic or based on a problem decomposition strategy, is described in [20]. Considering the monolithic version, which is computationally the worst, the search starts from the given initial system state, which is assigned the initial index of the index space, $\Im_0$. The complexity of the algorithm is given by the number of visited states (which equals the number of processed transitions). The behavior reconstruction mechanism has to generate all the other states to be assigned the same index, which are all the system states reachable from the initial state through silent paths. Let's denote by T the (maximum) total number of states reachable from a system state through silent paths. Denoting by I_0 the (upper bound of the) number of behavior states sharing index $\Im_0$, we have $I_0 = T + 1$. Each of such states has as successor states (if any) in the behavior graph a state whose assigned index is an index having depth 1 in the index space. Calling I_1 the (upper bound of the) number of behavior states having an index whose depth is 1, the following equality holds

$$I_1 = I_0 \cdot \beta \cdot n_{\mathrm{to}} \cdot (T + 1) = \beta \cdot n_{\mathrm{to}} \cdot (T + 1)^2 \tag{A.1}$$

where β is the branching factor of the index space, that is, the (maximum) number of edges exiting from a node in the index space, and n_{to} is the (maximum) number of transitions

leaving the same component state that generate the same message (according to the current viewer). In fact, I_1 accounts both for the $I_0 \cdot \beta \cdot n_{\text{to}}$ states reachable from the previous ones (the I_0 states having index $\Im_0$) through an observable transition, and in turn the states reachable from them through silent paths, which amounts (at most) to T for each such states and therefore are $I_0 \cdot \beta \cdot n_{\text{to}} \cdot T$ altogether. Following the same line of reasoning, the (upper bound of the) number of behavior states having an index whose depth is 2 is $I_2 = I_1 \cdot \beta \cdot n_{\text{to}} \cdot (T+1) = (\beta \cdot n_{\text{to}})^2 \cdot (T+1)^3$, and, in general, the (upper bound of the) number of behavior states having an index whose depth is i is

$$I_i = (T+1) \cdot \big[\beta \cdot n_{\text{to}} \cdot (T+1)\big]^i. \tag{A.2}$$

The total number of states produced by the behavior reconstruction algorithm is

$$N = \sum_{i=0}^{D} I_i = (T+1) \cdot \sum_{i=0}^{D} \big[\beta \cdot n_{\text{to}} \cdot (T+1)\big]^i, \tag{A.3}$$

where D is the depth of the index space. This sum gives a threefold result, depending on some conditions on the three non-negative integer variables β, n_{to} and T, as explained in the following:

$$N = \begin{cases} T+1 & \text{if } n_{\text{to}} = 0 \text{ or } \beta = 0, \\ (T+1) \cdot (D+1) = D+1 & \text{if } n_{\text{to}} = 1,\ \beta = 1,\ T = 0, \\ (T+1) \cdot \left(\frac{[\beta \cdot n_{\text{to}} \cdot (T+1)]^{D+1} - 1}{\beta \cdot n_{\text{to}} \cdot (T+1) - 1}\right) & \\ \quad = \mathrm{O}([\beta \cdot n_{\text{to}} \cdot T]^D) & \text{otherwise.} \end{cases} \tag{A.4}$$

Note how $n_{\text{to}} = 0$ holds only if the viewer is blind, while $n_{\text{to}} = 1$ holds if all the observable transitions exiting from a component state produce distinct messages, and $n_{\text{to}} > 1$ holds if the same message may be produced by several observable transitions leaving a component state.

Condition $\beta = 0$ holds if the index space includes index $\Im_0$ only, that is, the observation is empty; $\beta = 1$ holds if the index space is linear, that is, if the observation is certain (which is a particular case of an uncertain observation) while $\beta > 1$ holds if the observation is actually affected by uncertainty.

Condition $T = 0$ holds if all the transitions of all components are observable, that is, the system is completely observable, while $T > 1$ holds if there is at least a silent component transition throughout the system.

Notice how either $n_{\text{to}} = 0$ or $\beta = 0$ implies $D = 0$, which reinforces the first result in Eq. (A.4). The third case in (A.4) features the worst complexity. In order to finalize the analysis, the cost of T has to be estimated. Based on the definition of T, the following equality holds:

$$T = \sum_{j=1}^{p} (n_{\text{st}})^j = \begin{cases} 0 & \text{if } n_{\text{st}} = 0, \\ p & \text{if } n_{\text{st}} = 1, \\ \left(\frac{n_{\text{st}}^{p+1} - 1}{n_{\text{st}} - 1}\right) - 1 = \mathrm{O}(n_{\text{st}}^p) & \text{otherwise,} \end{cases} \tag{A.5}$$

where

- p is the (maximum) length of a silent path of the system, an upper bound of its being $\sum_{C\in\mathbb{C}} p_C$, i.e., the sum, encompassing all components, of the length of the (maximum) sequence of silent transitions of a single component, where the last transition in the sequence may loop back to the component state from which the first exits;[31]
- n_{st} is the (maximum) number of silent transitions leaving a system state, an upper bound of its being $\sum_{C\in\mathbb{C}} n_C$, where n_C is the maximum number of silent transitions leaving a state of the component model of C.

Thus, the third case in Eq. (A.4) becomes, in the worst case,

$$N = \mathrm{O}\big(\big[\beta \cdot n_{\mathrm{to}} \cdot n_{\mathrm{st}}^{p}\big]^{D}\big) \tag{A.6}$$

which is exponential in the product $p \cdot D$. □

Proof of Proposition 6. (*Sketch*) The time complexity of Algorithm 1 is dominated by the *for* cycle at lines 4–9, which is run for each state α in $\mathbf{S}^{\mathrm{a}}$. The number of states in $\mathbf{S}^{\mathrm{a}}$ is $\mathrm{O}(|\mathbf{S}| \cdot 2^{|\mathcal{R}|})$, where the cardinality of the powerset of $\mathcal{R}$ is the maximum number of distinct diagnoses given the current ruler. Each time the *for* cycle is run, it produces a number of iterations which equals the number of transitions exiting from the state β in B that makes up α. Therefore, the total number of iterations of the *for* cycle is $\mathrm{O}(f)$, where f is the maximum number of transitions exiting from a state in B. In conclusion, the algorithm is $\mathrm{O}(|\mathbf{S}| \cdot 2^{|\mathcal{R}|} \cdot f)$, which is exponential in the cardinality of the set of faulty transitions defining the ruler.

This upper bound is, however, very pessimistic and does not reflect the average case. In fact, in the above analysis it is implicitly assumed that (i) every subset of the set of all faults can be a diagnosis, and (ii) every such diagnosis can be inherent to whichever state in B. Both such assumptions are in general physically impossible given the current B as (i) only a set of faults corresponding to the set of faulty transitions in a behavioral path is a diagnosis, and (ii) such a diagnosis is inherent only to the states in B reachable from the root through a path characterized by this set of faults. Besides, the *for* cycle produces no iteration in case β is a leaf state of B, whereas in the formula f iterations are assumed.

In particular, in two cases the complexity of Algorithm 1 is linear, namely $\Theta(|\mathbf{T}|)$, since, indeed, the abduction $\mathit{Abd}(B, \mathcal{R})$ generated as output is isomorphic to the behavior B taken as input. This applies when B is a tree as well as when B is a graph wherein all the paths (either cyclic or acyclic) entering the same state are characterized by the same set of faults. The latter is the case, for instance, of $\mathit{Bhv}(\wp(\xi))$ displayed in Fig. 10 (Example 15, p. 254), whose relevant abduction $\mathit{Abd}(\wp(\xi))$ is shown in Fig. 11 (p. 255). □

Proof of Proposition 7. (*Sketch*) A candidate diagnosis of $\wp(\psi)$ is defined as $\delta = h \otimes \mathcal{R}$, where $h \otimes \mathcal{V} \in \mathit{Lang}(\mathit{Isp}(\mathcal{O}))$ and $h \in \mathit{Lang}(\mathit{Bhv}(\psi, \psi_0))$. Similarly, a candidate diagnosis of $\wp'(\psi)$ is defined as $\delta' = h \otimes \mathcal{R}'$, where $h \otimes \mathcal{V} \in \mathit{Lang}(\mathit{Isp}(\mathcal{O}))$ and $h \in \mathit{Lang}(\mathit{Bhv}(\psi, \psi_0))$. As such, δ and δ' differ in the by-product with the ruler only, as

[31] Including possibly looping-back silent transitions is right to the point for complexity analysis since this way also the effort for processing possible cycles (necessarily linking states having the same index) is accounted for.

the two conditions on h are identical. Since $\delta = h \otimes \mathcal{R} = \{\varphi \mid (T, \varphi) \in \mathcal{R}, T \in h\}$, we have $\delta_{[\mathcal{R}']} = \{\varphi' \mid \varphi \in \delta, \varphi' = Ren(\varphi, \mathcal{R}, \mathcal{R}')\} = \{\varphi' \mid (T, \varphi') \in \mathcal{R}', T \in h\} = \delta'$. □

Proof of Lemma 1. (*Sketch*) Eq. (37) is supported by the fact that $Lang(\mathcal{M}) = \{h \otimes \mathcal{V} \mid h \in Bhv(\psi, \hat{\psi}_0)\}$. Hence, $Lang(\mathcal{M}_{[\hat{\mathcal{V}}]}) = \{h \otimes \hat{\mathcal{V}} \mid h \in Bhv(\hat{\psi}, \hat{\psi}_0)\}$, and $Lang(\mathcal{M}_{[\hat{\mathcal{V}}]} \asymp \hat{\mathcal{O}}) = \{h \otimes \hat{\mathcal{V}} \mid h \in Bhv(\hat{\psi}, \hat{\psi}_0), h \otimes \hat{\mathcal{V}} \in Lang(Isp(\hat{\mathcal{O}}))\}$. Thus, $\Delta(Lang(\mathcal{M}_{[\hat{\mathcal{V}}]} \asymp \hat{\mathcal{O}})) = \{\delta \mid \delta = h \otimes \hat{\mathcal{R}}, h \in Bhv(\hat{\psi}, \hat{\psi}_0), h \otimes \hat{\mathcal{V}} \in Lang(Isp(\hat{\mathcal{O}}))\} = \Delta(\hat{\wp}(\hat{\psi}))$. □

Proof of Lemma 2. (*Sketch*) $Lang(\mathcal{M}) = \{h \otimes \hat{\mathcal{V}} \mid h \in Bhv(\wp(\psi)\}$, $Lang(\mathcal{M} \asymp \hat{\mathcal{O}}) = \{h \otimes \hat{\mathcal{V}} \mid h \in Bhv(\wp(\psi)), h \otimes \hat{\mathcal{V}} \in Lang(Isp(\hat{\mathcal{O}}))\}$. Hence, $\Delta(Lang(\mathcal{M} \asymp \hat{\mathcal{O}})) = \{\delta \mid \delta = h \otimes \hat{\mathcal{R}}, h \in Bhv(\wp(\psi)), h \otimes \hat{\mathcal{V}} \in Lang(Isp(\hat{\mathcal{O}}))\}$. Based on Proposition 3, $Bhv(\wp(\psi)) \ni Bhv(\hat{\wp}(\hat{\psi}))$. Thus, $\Delta(Lang(\mathcal{M} \asymp \hat{\mathcal{O}})) \subseteq \Delta(\hat{\wp}(\hat{\psi}))$. This inclusion (soundness) is in fact an equality (completeness), as $\nexists h(h \in (Bhv(\psi, \psi_0) - Bhv(\wp(\psi))), h \otimes \hat{\mathcal{V}} \in Lang(Isp(\hat{\mathcal{O}})))$. □

Proof of Proposition 8. (*Sketch*) Based on Lemma 1, $\Delta(\mathcal{M}_{[\hat{\mathcal{V}}]} \asymp \hat{\mathcal{O}}) = \Delta(\hat{\wp}'(\hat{\psi}))$, where $\hat{\wp}'(\hat{\psi}) = (\hat{\psi}_0, \hat{\mathcal{V}}, \hat{\mathcal{O}}, \mathcal{R}, \hat{\mathcal{K}})$. Thus, Eq. (39) derives from Proposition 7. □

Proof of Proposition 9. (*Sketch*) Based on Lemma 2, $\Delta(\mathcal{M}_{[\hat{\mathcal{V}}]} \asymp \hat{\mathcal{O}}) = \Delta(\hat{\wp}'(\hat{\psi}))$, where $\hat{\wp}'(\hat{\psi}) = (\hat{\psi}_0, \hat{\mathcal{V}}, \mathcal{O}, \mathcal{R}, \mathcal{K})$. Thus, Eq. (40) derives from Proposition 7. □

Proof of Proposition 10. (*Sketch*) The generation of the index space performed at line 1 of Algorithm 2 is exponential in the number of nodes of the observation graph, as proven in [20]. The time complexity of the remaining statements is dominated by the *for* cycle at lines 6–13, which is run once for each edge in the index space leaving a node $\Im$, where $\Im$ makes up a state in $\breve{\mathbb{S}}$. Owing to the determinism of the map, the maximum number of states $\breve{\mu}$ in $\breve{\mathbb{S}}$ sharing $\Im$ is given by the number of distinct paths reaching $\Im$ from $\Im_0$ in the index space. Thus, an upper bound for the complexity is given by the sum of all the paths from $\Im_0$ to whichever node of the index space, which equals the sum of the number of transitions along whichever path from $\Im_0$ to a final node in the index space (which equals the sum of the lengths of all the observation instances complying with the observation index). Dually, another upper bound of the same complexity is given by the number of transitions belonging to the considered map up to a depth that equals the depth D of the index space. In both cases this upper bound is formally given by

$$\sum_{i=1}^{D} \beta^i = \begin{cases} D & \text{if } \beta = 1, \\ \left(\frac{\beta^{D+1}}{\beta^{-1}}\right) - 1 & \text{otherwise,} \end{cases} \tag{A.7}$$

where β is the branch parameter in either the index space or the map (the smaller value gives a better estimation). Since D equals the number of nodes of the observation graph, we can conclude that we have found an upper bound of the time complexity of Algorithm 2 which is exponential in the number of nodes of the observation graph. □

Proof of Lemma 3. (*Sketch*) Since $\mathcal{V} \Subset \hat{\mathcal{V}}$, $\mathcal{O} \ni \hat{\mathcal{O}}_{[\mathcal{V}]}$, based on Proposition 3, we have $Bhv(\wp(\psi)) \ni Bhv(\hat{\wp}(\hat{\psi}))$. Thus, $Lang(A \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})) = \{h \mid h \in Lang(Bhv(\wp(\psi))), h \otimes$

$\hat{\mathcal{V}} \in Lang(Isp(\hat{\mathcal{O}}))\}$, and $\Delta(A \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})) = \{\delta \mid \delta = h \otimes \hat{\mathcal{R}}, h \in Lang(Bhv(\wp(\psi))), h \otimes \hat{\mathcal{V}} \in Lang(Isp(\hat{\mathcal{O}}))\} \subseteq \Delta(\hat{\wp}(\hat{\psi}))$. This inclusion (soundness) is in fact an equality (completeness), as $\nexists h(h \in (Bhv(\psi, \psi_0) - Bhv(\wp(\psi))), h \otimes \hat{\mathcal{V}} \in Lang(Isp(\hat{\mathcal{O}})))$. □

Proof of Proposition 11. (*Sketch*) Based on Lemma 3, $\Delta(A \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})) = \Delta(\hat{\wp}'(\hat{\psi}))$, where $\hat{\wp}'(\hat{\psi}) = (\hat{\psi}_0, \hat{\mathcal{V}}, \hat{\mathcal{O}}, \mathcal{R}, \hat{\mathcal{K}})$. Thus, Eq. (43) derives from Proposition 7. □

Proof of Proposition 12. (*Sketch*) Since $\mathcal{V} \Subset \hat{\mathcal{V}}$, $\mathcal{O} \ni \hat{\mathcal{O}}_{[\mathcal{V}]}$, based on Proposition 3, we have $Bhv(\wp(\psi)) \ni Bhv(\hat{\wp}(\hat{\psi}))$. Thus, $Lang(B \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})) = \{h \mid h \in Lang(Bhv(\wp(\psi))), h \otimes \hat{\mathcal{V}} \in Lang(Isp(\hat{\mathcal{O}}))\}$, and, due to Proposition 4, $\Delta(Abd((B \asymp (\hat{\mathcal{O}}, \hat{\mathcal{V}})), \hat{\mathcal{R}})) = \{\delta \mid \delta = h \otimes \hat{\mathcal{R}}, h \in Lang(Bhv(\wp(\psi))), h \otimes \hat{\mathcal{V}} \in Lang(Isp(\hat{\mathcal{O}}))\} \subseteq \Delta(\hat{\wp}(\hat{\psi}))$. This inclusion is in fact an equality, as $\nexists h(h \in (Bhv(\psi, \psi_0) - Bhv(\wp(\psi))), h \otimes \hat{\mathcal{V}} \in Lang(Isp(\hat{\mathcal{O}})))$. □

Proof of Proposition 13. (*Sketch*) The *Join* operator resembles the *Merger* algorithm introduced in [10], whose asymptotic upper bound is exponential in the maximum 'extent of the routes' resulting from the merging. According to the terminology in [10], a route is the subgraph of the behavior space traversed by a history and its extent is the number of edges incorporated in it, therefore the maximum extent of the routes corresponds, in the current paper, to the depth of the behavior graph created by a *Join* operation. This depth can be estimated as $p \cdot D$, where p is the maximum length of a silent path of the system, given the current viewer, and D is the depth of the index space of the considered problem $\wp(\psi)$. This mirrors the result of the complexity analysis of the monolithic behavior reconstruction method discussed in Section 8. However, while there the upper bound of p can be computed based on the features of all the (say m) component models, here it is given by the features of the n (with $m \geqslant n$) virtual component models B_i. Therefore, the expected value of p is smaller when dealing with fragmented problems, since each behavior B_i embodies the interface constraints between the components in ψ_i and possibly also constraints inherent to the observation. As a consequence, the length of the system silent paths computed as the sum of the (longest) silent paths of all B_i is a better (i.e. closer to reality) estimate than the previous one, which does not take into account any such constraints. The same applies to all the other parameter values, that is, both the model-based approach and the fragmented approach are exponential in nature, however, the latter reduces the input values of the complexity function with respect to the former, which translates in reducing the value of the exponent as well as of the basis. □

Proof of Proposition 14. (*Sketch*) Eq. (60) can be simply proven by induction on the nodes of the diagnostic tree. According to Eq. (59), it is trivially proved when N is a leaf node (basis). The inductive step (when N is an internal node) is grounded on the definition of *Join*, specifically, on Eq. (47). □

Proof of Proposition 15. (*Sketch*) Based on Proposition 14, Eq. (64) translates to $Bhv(\wp(\psi)) \doteq Bhv(\wp(\psi'))$. Let $h = \langle T_1, \ldots, T_n \rangle \in Lang(Bhv(\wp(\psi)))$. Consider $h' = \langle T'_1, \ldots, T'_n \rangle$ where $\forall i \in [1..n]\,(T_i \between T'_i)$. We show that $h' \in Lang(Bhv(\wp(\psi')))$. In fact, based on Eq. (63), the initial states of $Bhv(\wp(\psi))$ and $Bhv(\wp(\psi'))$ are $S_0 = (\sigma_0, \Im_0)$

and $S'_0 = (\sigma'_0, \Im'_0)$, respectively, where $\sigma_0 = \sigma'_0$ (basis). Since $\mathcal{O} \doteq \mathcal{O}'$, the relevant index spaces $Isp(\mathcal{O})$ and $Isp(\mathcal{O}')$ match in both nodes and edges, where each corresponding edge is marked by corresponding labels in $\mathcal{V}$ and $\mathcal{V}'$. Thus, it suffices to show that if $(\sigma_{i-1}, \Im_{i-1}) \xrightarrow{T_i} (\sigma_i, \Im_i)$ is the ith transition in h and $\sigma_{i-1} = \sigma'_{i-1}$, $\Im_{i-1} \between \Im'_{i-1}$, then $(\sigma'_{i-1}, \Im'_{i-1}) \xrightarrow{T'_i} (\sigma'_i, \Im'_i)$ is the ith transition in h', where $\sigma'_i = \sigma_i$ and $\Im'_i \between \Im_i$ (induction). In fact, based on the definition of behavior given in Section 8, $\sigma_{i-1} \xrightarrow{T_i} \sigma_i$ is a transition in $Bhv(\psi, \psi_0)$; besides, if $(T_i, \ell) \in \mathcal{V}$ then $\Im_{i-1} \xrightarrow{\ell} \Im_i \in Lang(Isp(\mathcal{O}))$ else $\Im_i = \Im_{i-1}$. These properties support the following: $\sigma'_{i-1} \xrightarrow{T'_i} \sigma_i$ is a transition in $Bhv(\psi', \psi_0)$, where $T'_i \between T_i$ and $\sigma'_i = \sigma'$; besides, if $(T'_i, \ell') \in \mathcal{V}'$ then $\Im'_{i-1} \xrightarrow{\ell'} \Im'_i \in Lang(Isp(\mathcal{O}'))$ else $\Im'_i = \Im'_{i-1}$, where $T'_i \between T_i$, $\ell \between \ell'$, and $\Im'_i \between \Im_i$. Hence, T'_i moves ψ' to a new state $S'_i = (\sigma'_i, \Im'_i)$ that is isomorphic to the state reached by ψ by means of transition T_i. By induction on the transitions of h, if $h \in Lang(Bhv(\wp(\psi)))$ then $h' \in Lang(Bhv(\wp(\psi')))$. The vice versa (if $h' \in Lang(Bhv(\wp(\psi')))$ then $h \in Lang(Bhv(\wp(\psi)))$) holds owing to the symmetry of the isomorphism relationship. □

Proof of Proposition 16. (*Sketch*) Eqs. (70)–(71) force D^{g} to embody a node picked up from each part $\mathbb{N} \in \aleph^{\mathrm{t}}$. Eqs. (72)–(73) make each node in D^{g} to have the same number of leaving edges as in D^{t}, where information on child nodes in D^{t} (identifiers of the subsystems) are maintained by the labels marking the corresponding edges in D^{g}. Eq. (76) can be proved by induction on nodes of D^{g}. Based on a comparison between Eq. (74) and Eq. (59), we have to show that $\mathbb{B}$ is the same in both equations, as the basis of the proof (N leaf node) is identical, as well as the arguments $\mathcal{V}$ and $\mathcal{O}$ of *Join*. Hence, in the induction step we assume that Eq. (76) is met for the child nodes of N. Considering Eq. (75), based on Eq. (72), $N \xrightarrow{\psi''} N' \in \mathbf{E}^{\mathrm{g}}$ corresponds to $N \to N'' \in \mathbf{E}^{\mathrm{t}}$, where $N' \doteq N''$. Eq. (73) makes such correspondence complete, that is, each label ψ'' marking the edges leaving N in D^{g} identifies the subsystem relevant to a child node N'' of N in D^{t}. According to Eq. (75), two cases are possible. If $\psi'' = \psi'$, then $B' = Bhv^{\mathrm{g}}(N')$, where N' coincides with N'' in D^{t}, thereby (by assumption of the inductive step), $B' = Bhv^{\mathrm{t}}(N')$. Instead, if $\psi'' \neq \psi'$, then $Lang(Bhv^{\mathrm{g}}(N')) = Lang(Bhv^{\mathrm{t}}(N'))$ (by assumption), where $N' \doteq N''$, hence (owing to Proposition 15) $Bhv^{\mathrm{t}}(N') \doteq Bhv^{\mathrm{t}}(N'')$. Therefore, $B' = (Bhv^{\mathrm{g}}(N'))_{[\psi'']}$, $Lang(B') = Lang(Bhv^{\mathrm{t}}(N''))$. □

Proof of Proposition 17. (*Sketch*) Eq. (80) is grounded on Corollary 16.2, $Lang(Bhv^{\mathrm{g}}(N_0)) = Lang(Bhv(\hat{\wp}(\psi)))$, and on Proposition 4. □

References

[1] C. Cassandras, S. Lafortune, Introduction to Discrete Event Systems, The Kluwer International Series in Discrete Event Dynamic Systems, vol. 11, Kluwer Academic Publisher, Boston, MA, 1999.

[2] G. Lamperti, P. Pogliano, Event-based reasoning for short circuit diagnosis in power transmission networks, in: Proc. Fifteenth International Joint Conference on Artificial Intelligence (IJCAI'97), Nagoya, Japan, 1997, pp. 446–451.

[3] Y. Pencolé, Decentralized diagnoser approach: application to telecommunication networks, in: Proc. Eleventh International Workshop on Principles of Diagnosis (DX'00), Morelia, MX, 2000, pp. 185–192.
[4] M. Cordier, C. Largouët, Using model-checking techniques for diagnosing discrete-event systems, in: Proc. Twelfth International Workshop on Principles of Diagnosis (DX'01), San Sicario, Italy, 2001, pp. 39–46.
[5] Y. Pencolé, M. Cordier, L. Rozé, Incremental decentralized diagnosis approach for the supervision of a telecommunication network, in: Proc. Twelfth International Workshop on Principles of Diagnosis (DX'01), San Sicario, Italy, 2001, pp. 151–158.
[6] L. Console, C. Picardi, M. Ribaudo, Process algebras for systems diagnosis, Artificial Intelligence 142 (1) (2002) 19–51.
[7] S. Zad, R. Kwong, W. Wonham, Fault diagnosis in timed discrete-event systems, in: Proc. American Control Conference, Pheonix, AZ, 1999, pp. 1756–1761.
[8] J. Lunze, Diagnosis of quantized systems based on a timed discrete-event model, IEEE Trans. Systems Man Cybernet. Part A: Systems and Humans 30 (3) (2000) 322–335.
[9] S. McIlraith, Explanatory diagnosis: conjecturing actions to explain observations, in: Proc. Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98), Trento, Italy, 1998, pp. 167–177.
[10] P. Baroni, G. Lamperti, P. Pogliano, M. Zanella, Diagnosis of large active systems, Artificial Intelligence 110 (1) (1999) 135–183.
[11] C. Barral, S. McIlraith, T. Son, Formulating diagnostic problem solving using an action language with narratives and sensing, in: Proc. Seventh International Conference on Knowledge Representation and Reasoning (KR'2000), Breckenridge, CO, 2000, pp. 311–322.
[12] L. Console, C. Picardi, M. Ribaudo, Diagnosis and diagnosability using PEPA, in: Proc. Fourteenth European Conference on Artificial Intelligence (ECAI'2000), Berlin, Germany, 2000, pp. 131–135.
[13] M. Cordier, A. Grastien, C. Largouët, Y. Pencolé, Efficient trajectories computing exploiting inversibility properties, in: Proc. Fourteenth International Workshop on Principles of Diagnosis (DX'03), Washington DC, 2003, pp. 93–98.
[14] L. Rozé, Supervision of telecommunication network: a diagnoser approach, in: Proc. Eighth International Workshop on Principles of Diagnosis (DX'97), Mont St. Michel, France, 1997, pp. 103–111.
[15] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, D. Teneketzis, Diagnosability of discrete-event systems, IEEE Trans. Automat. Control 40 (9) (1995) 1555–1575.
[16] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, D. Teneketzis, Failure diagnosis using discrete-event models, IEEE Trans. Control Syst. Technol. 4 (2) (1996) 105–124.
[17] J. Kurien, P. Nayak, Back to the future for consistency-based trajectory tracking, in: Proc. Eleventh International Workshop on Principles of Diagnosis (DX'00), Morelia, MX, 2000, pp. 92–100.
[18] P. Baroni, G. Lamperti, P. Pogliano, M. Zanella, Diagnosis of a class of distributed discrete-event systems, IEEE Trans. Systems Man Cybernet. Part A: Systems and Humans 30 (6) (2000) 731–752.
[19] G. Lamperti, M. Zanella, Diagnosis of Active Systems—Principles and Techniques, The Kluwer International Series in Engineering and Computer Science, vol. 741, Kluwer Academic Publisher, Dordrecht, NL, 2003.
[20] G. Lamperti, M. Zanella, Diagnosis of discrete-event systems from uncertain temporal observations, Artificial Intelligence 137 (1–2) (2002) 91–163.
[21] A. Aho, R. Sethi, J. Ullman, Compilers—Principles, Techniques, and Tools, Addison-Wesley, Reading, MA, 1986.
[22] A. Silberschatz, H. Korth, S. Sudarshan, Database System Concepts, McGraw-Hill, New York, 1998.
[23] G. Lamperti, M. Zanella, Principles of distributed diagnosis of discrete-event systems, in: Proc. Twelfth International Workshop on Principles of Diagnosis (DX'01), San Sicario, Italy, 2001, pp. 95–102.
[24] G. Lamperti, M. Zanella, EDEN: An intelligent software environment for diagnosis of discrete-event systems, Appl. Intelligence—The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies 18 (2003) 55–77.
[25] P. Laborie, J. Krivine, Automatic generation of chronicles and its application to alarm processing in power distribution systems, in: Proc. Eighth International Workshop on Principles of Diagnosis (DX'97), Mont St. Michel, France, 1997.
[26] L. Rozé, M. Cordier, Diagnosing discrete-event systems: extending the 'diagnoser approach' to deal with telecommunication networks, J. Discrete Event Dynamic Syst. 12 (2002) 43–81.

[27] G. Lamperti, M. Zanella, Diagnosis of discrete-event systems integrating synchronous and asynchronous behavior, in: Proc. Tenth International Workshop on Principles of Diagnosis (DX'99), Loch Awe, UK, 1999, pp. 129–139.
[28] M. Cordier, C. Dousson, Alarm driven monitoring based on chronicles, in: Proc. Fourth IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS'2000), Budapest, Hungary, 2000, pp. 286–291.
[29] A. Osmani, F. Lévy, A constraint-based approach to simulate faults in telecommunication networks, in: Intelligent Problem Solving. Methodologies and Approaches: Proc. 13th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE 2000, New Orleans, LA, June 2000, in: Lecture Notes in Computer Science, vol. 1821/2000, Springer, Berlin, 2000, pp. 463–474.